End to End Travel Management in The Tourism Sector Using Web Development, IoT, ML

submitted by

Abhijit Mandal
Department: Computer Science and Engineering
College Name: Jadavpur University, West Bengal

Arghya Nayek
Department: Information Technology
College Name: Government College of Engineering and Textile Technology, Serampore

Hriya Ghosh
Department: Computer Science and Engineering
College Name: Government College of Engineering and Leather Technology, West Bengal

Pervez Mosaraf SK
Department: Electrical Engineering
College Name: Jadavpur University, West Bengal

Raktim Das
Department: Computer Science and Engineering
College Name: Government College of Engineering and Textile Technology, Berhampore

Shritama Ghosh
Department: Electronics and Communication Engineering
College Name: Ramkrishna Mahato Government Engineering College, West Bengal

SK Nisar Mehtaz Khan
Department: Mechanical Engineering
College Name: Government College of Engineering and Textile Technology, Berhampore

Sujit Kumar Parida
Department: Electrical Engineering
College Name: College of Engineering and Technology, Bhubaneswar

under the guidance of

Mr. Asit Deva
Mr. Hemant Baid
Mr. Soumen De
Mr. Raveendran Vasudevan

Year: 2020

Acknowledgement

The Internship opportunity we have with Siemens was a great chance for learning and professional development. Therefore, we consider ourselves very happy individuals. We are also grateful for having a chance to meet so many wonderful people and personalities who helped and guided us enough during this internship period.

I would like to express my sincere gratitude to my supervisors Mr. Asit Deva,Mr. Hemant Baid, Mr. Soumen De, Mr. Raveendran Vasudevan for providing their invaluable guidance, comments and suggestions throughout the course of the project.

I would specially thank Mr. Asit Deva for constantly motivating me to work harder and for referring the required lessons and suggestions and Mr. Hemant Baid for providing me an overview of the documentation and other necessary details.

Also, I would like to thank Mr. Soumen De for his assistance and Mr. Raveendran Vasudevan for his help during the preparation of the project.

My sincere thanks also go to all the representatives of Siemens Scholarship Program for their kind cooperation in all spheres during my project work.

Last but not the least, I thank my parents and all my family members for their support and motivation they have provided throughout my project.

Abhijit Mandal
Arghya Nayek
Hriya Ghosh
Pervez Mosaraf SK
Raktim Das
Shritama Ghosh
SK Nisar Mehtaz Khan
Sujit Kumar Parida

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. We understand that any violation of the above will be cause for disciplinary action and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Date:

Abhijit Mandal
Arghya Nayek
Hriya Ghosh
Pervez Mosaraf SK
Raktim Das
Shritama Ghosh
SK Nisar Mehtaz Khan
Sujit Kumar Parida

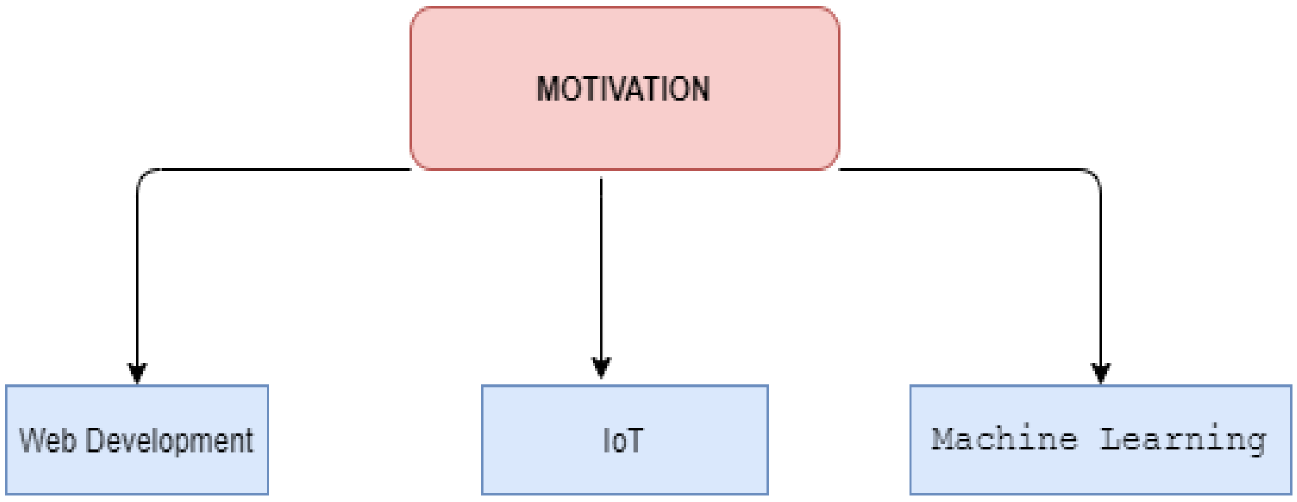
Contents

1.Introduction

1.1. Background

Stay home, stay safe- these words that have grounded holiday plans the world over as the novel coronavirus pandemic rages. Tourism is among the hardest hit of all sectors by the COVID-19 pandemic. The multibillion-dollar tourism sector, a growth engine in many economies, stares at gigantic losses. Amid uncertainty to boost the tourism sector needs active management and calculated steps. So, normalcy in the tourism sector can be achieved with Industry 4.0 technology. It takes almost 14 to 21 days of self-isolation to show the acute symptoms. At this time, we can monitor the health parameters of large numbers of people with very few medical personnel and also can detect the possibility of Covid-19 using software-based detectors through which we can easily test the possibility of getting infected within a few minutes and at no cost.

1.2. Motivation



<u>Web Development:</u>

Web development is the work involved in developing a Website for the Internet (World Wide Web) or an intranet (a private network). Web development can range from developing a simple single static page of plain text to complex Web-based Internet applications (Web apps), electronic businesses, and social network services.

We have used HTML, CSS, JavaScript, MySQL, PHP to develop web applications.

- **HTML** –

Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by *tags*, written using angle brackets. Tags such as <**img** /> and <**input** /> directly introduce content into the page. Other tags such as <**p**> surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behaviour and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.

- **CSS** –

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML.CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

CSS is designed to enable the separation of presentation and content, including layout, colours, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device.

The name *cascading* comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

- **PHP –**

PHP is a general-purpose scripting language especially suited to web development. It was originally created by Danish-Canadian programmer Rasmus Lerdorf in 1994; the PHP reference implementation is now produced by The PHP Group. PHP originally stood for *Personal Home Page*, but it now stands for the recursive initialism *PHP: Hypertext Pre-processor*.

PHP code is usually processed on a web server by a PHP interpreter implemented as a module, a daemon or as a Common Gateway Interface (CGI) executable. On a web server, the result of the interpreted and executed PHP code – which may be any type of data, such as generated HTML or binary image data – would form the whole or part of a HTTP response. Various web template systems, web content management systems, and web frameworks exist which can be employed to orchestrate or facilitate the generation of that response. Additionally, PHP can be used for many programming tasks outside of the web context, such as standalone graphical applications] and robotic drone control Arbitrary PHP code can also be interpreted and executed via command-line interface (CLI).

The standard PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

The PHP language evolved without a written formal specification or standard until 2014, with the original implementation acting as the *de facto* standard which other implementations aimed to follow. Since 2014, work has gone on to create a formal PHP specification.

- **JavaScript –**

JavaScript often abbreviated as **JS**, is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions.

Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web.[JavaScript enables interactive web pages and is an essential part of web applications. The vast majority of websites use it for client-side page behaviour, and all major web browsers have a dedicated JavaScript engine to execute it.

As a multi-paradigm language, JavaScript supports event-driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM). However, the language itself does not include any input/output (I/O), such as networking, storage, or graphics facilities, as the host environment (usually a web browser) provides those APIs.

JavaScript engines were originally used only in web browsers, but they are now embedded in some servers, usually via Node.js. They are also embedded in a variety of applications created with frameworks such as Electron and Cordova.

- **MySQL** –

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. A relational database organizes data into one or more data tables in which data types may be related to each other; these relations help structure the data. SQL is a language programmer use to create, modify and extract data from the relational database, as well as control user access to the database. In addition to relational databases and SQL, an RDBMS like MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups.

- **IoT** –

The **Internet of Things** (**IoT**) describes the network of physical objects—“things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet.

The definition of the Internet of Things has evolved due to the convergence of multiple technologies, real-time analytics, machine learning, commodity sensors, and embedded systems. Traditional fields of embedded systems, wireless sensor networks, control systems, automation (including home and building automation), and others all contribute to enabling the Internet of things. In the consumer market, IoT technology is most synonymous with products pertaining to the concept of the "smart home", including devices and appliances (such as lighting fixtures, thermostats, home security systems and cameras, and other home appliances) that support one or more common ecosystems, and can be controlled via devices associated with that ecosystem, such as smartphones and smart speakers.

The **Internet of Medical Things** (**IoMT**) is an application of the IoT for medical and health related purposes, data collection and analysis for research, and monitoring. The IoMT has been referenced as "Smart Healthcare", as the technology for creating a digitized healthcare system, connecting available medical resources and healthcare services.

IoT devices can be used to enable remote health monitoring and emergency notification systems. These health monitoring devices can range from blood pressure and heart rate monitors to advanced devices capable of monitoring specialized implants, such as pacemakers, Fitbit electronic wristbands, or advanced hearing aids. Some hospitals have begun implementing "smart beds" that can detect when they are occupied and when a patient is attempting to get up. It can also adjust itself to ensure appropriate pressure and support is applied to the patient without the manual interaction of nurses. A 2015 Goldman Sachs report indicated that healthcare IoT devices "can save the United States more than $300 billion in annual healthcare expenditures by increasing revenue and decreasing cost." Moreover, the use of mobile devices to support medical follow-up led to the creation of 'm-health', used analysed health statistics."

Specialized sensors can also be equipped within living spaces to monitor the health and general well-being of senior citizens, while also ensuring that proper treatment is being administered and assisting people regain lost mobility via therapy as well. These sensors create a network of intelligent sensors that are able to collect, process, transfer, and analyse valuable information in different environments, such as connecting in-home monitoring devices to hospital-based systems. Other consumer devices to encourage healthy living, such as connected scales or wearable heart monitors, are also a possibility with the IoT. End-to-end health monitoring IoT platforms are also available for antenatal and chronic patients, helping one manage health vitals and recurring medication requirements.

Advances in plastic and fabric electronics fabrication methods have enabled ultra-low cost, use-and-throw IoMT sensors. These sensors, along with the required RFID electronics, can be fabricated on paper or e-textiles for wireless powered disposable sensing devices. Applications have been established for point-of-care medical diagnostics, where portability and low system-complexity is essential.

IoMT in the healthcare industry is now permitting doctors, patients, and others, such as guardians of patients, nurses, families, and similar, to be part of a system, where patient records are saved in a database, allowing doctors and the rest of the medical staff to have access to patient information. Moreover, IoT-based systems are patient-centred, which involves being flexible to the patient's medical conditions. IoMT in the insurance industry provides access to better and new types of dynamic information. This includes sensor-based solutions such as biosensors, wearables, connected health devices, and mobile apps to track customer behaviour. This can lead to more accurate underwriting and new pricing models.

The application of the IoT in COVID-19 treatment and remote monitoring can be done through the connection of powerful wireless solutions. The connectivity enables health practitioners to capture patient's data and apply complex algorithms in health data analysis.

- **Machine Learning** –

Logistic Regression Machine Learning Algorithm is one of the most widely used algorithms to detect the possibility of any disease. Our project aims at predicting the overall probability of getting infected in COVID-19 disease.

The reason of using Logistic regression as the primary prediction algorithm in our project is

1. Simple algorithm: Logistic Regression is one of the simplest machine learning algorithms and is easy to implement yet provides great training efficiency. It doesn't require high computation power. It outputs 0/1 depending on positive or negative samples.

2. Well-calibrated: Logistic Regression outputs well-calibrated probabilities along with classification results.

Feature selection: Like linear regression, logistic regression does work better when you remove attributes that are unrelated to the output variable as well as attributes that are very similar (correlated) to each other. Therefore, Feature Engineering plays an important role in regards to the performance of Logistic and also Linear Regression

The reason of using Decision tree as a model to select important features:

1. Decision trees can perform feature selection or variable screening completely. They can work on both categorical and numerical data. Furthermore, they can handle problems with multiple results or outputs.
2. Decision tree uses an internal decision-making logic.
3. Based on the tree nodes and their level of presence one can determine which features are important in prediction.

We have used Python language and it’s sklearn module in various aspects of this project.

The motivation of using python language is

1. Simple and consistent: Python offers concise and readable code. While complex algorithms and versatile workflows stand behind machine learning and AI, Python’s simplicity allows developers to write reliable systems.

2. Availability of easily usable Libraries and Frameworks: Implementing AI and ML algorithms can be tricky and requires a lot of time. It’s vital to have a well-structured and well-tested environment to enable developers to come up with the best coding solutions. Python, with its rich technology stack, has an extensive set of libraries for artificial intelligence and machine learning like Numpy, Pandas,Sklearn,Seaborn.
3. Platform Independent Language: Platform independence refers to a programming language or framework allowing developers to implement things on one machine and use them on another machine without any (or with only minimal) changes.

1.3 Problem Statement

Tourism is among the hardest hit of all sectors by the COVID-19 pandemic. The multibillion-dollar tourism sector, a growth engine in many economies, stares at gigantic losses. Amid uncertainty to boost the tourism sector needs active management and calculated steps. So, normalcy in the tourism sector can be achieved with Industry 4.0 technology. It takes almost 14 to 21 days of self-isolation to show the acute symptoms.

So, with a web-based application that allows the hotel manager (Admin Corner) to handle all hotel activities online with a self-isolation facility. Interactive GUI and the ability to manage various hotel bookings and rooms make this system very flexible and convenient with Quarantine room booking, staff management and other necessary hotel management features. Again, within this time IoT based smart remote health monitoring can be a solution to monitor the health parameters when the whole world is facing COVID-19 pandemic. Wireless sensors are used to collect and transmit signals of interest and a processor is programmed to receive and automatically analyze the sensor signals. Using a single parameter monitoring system an approach to a remote health monitoring system was designed that extends healthcare from the hotels, traditional clinic or hospital setting to the patient's home.

Again, the availability of test kits is also not sufficient. So, to develop a detector using sensors and surveys through which millions of people can be tested with high accuracy and the probability of doctors to get infected is also slowed down as we are using contactless sensors and a small set of questions which can be asked using websites.

1.4 Objective

1.4.1 Objective of Web Application

During the past several decades personnel function has been transformed from a relatively obscure record keeping staff to central and top-level management function. There are many factors that have influenced this transformation like technological advances, professionalism, and general recognition of human beings as the most important resources.

- A computer-based management system is designed to handle all the primary information required to calculate monthly statements. Separate database is maintained to handle all the details required for the correct statement calculation and generation.
- This project intends to introduce more user friendliness in the various activities such as record updating, maintenance, and searching.
- The searching of records has been made quite simple as all the details of the customer can be obtained by simply keying in the identification of that customer.

- Similarly, record maintenance and updating can also be accomplished by using the identification of the customer with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.
- The entire information has been maintained in the database or Files and whoever wants to retrieve can't retrieve, only authorization users can retrieve the necessary information which can be easily be accessible from the file.
- The main objective of the entire activity is to automate the process of day to day activities of Hotel like:

1. Room booking for quarantine service
2. Admission of a New Customer,
3. Assign a room according to customer's demand,
4. Checkout of a computer and releasing the room
5. Advance online bookings.
6. Online Cancellation.
7. List of Regular customers.
8. Email facility.
9. Feedbacks This project has some more features:

1.4.2 Objective of IoT based health monitoring system

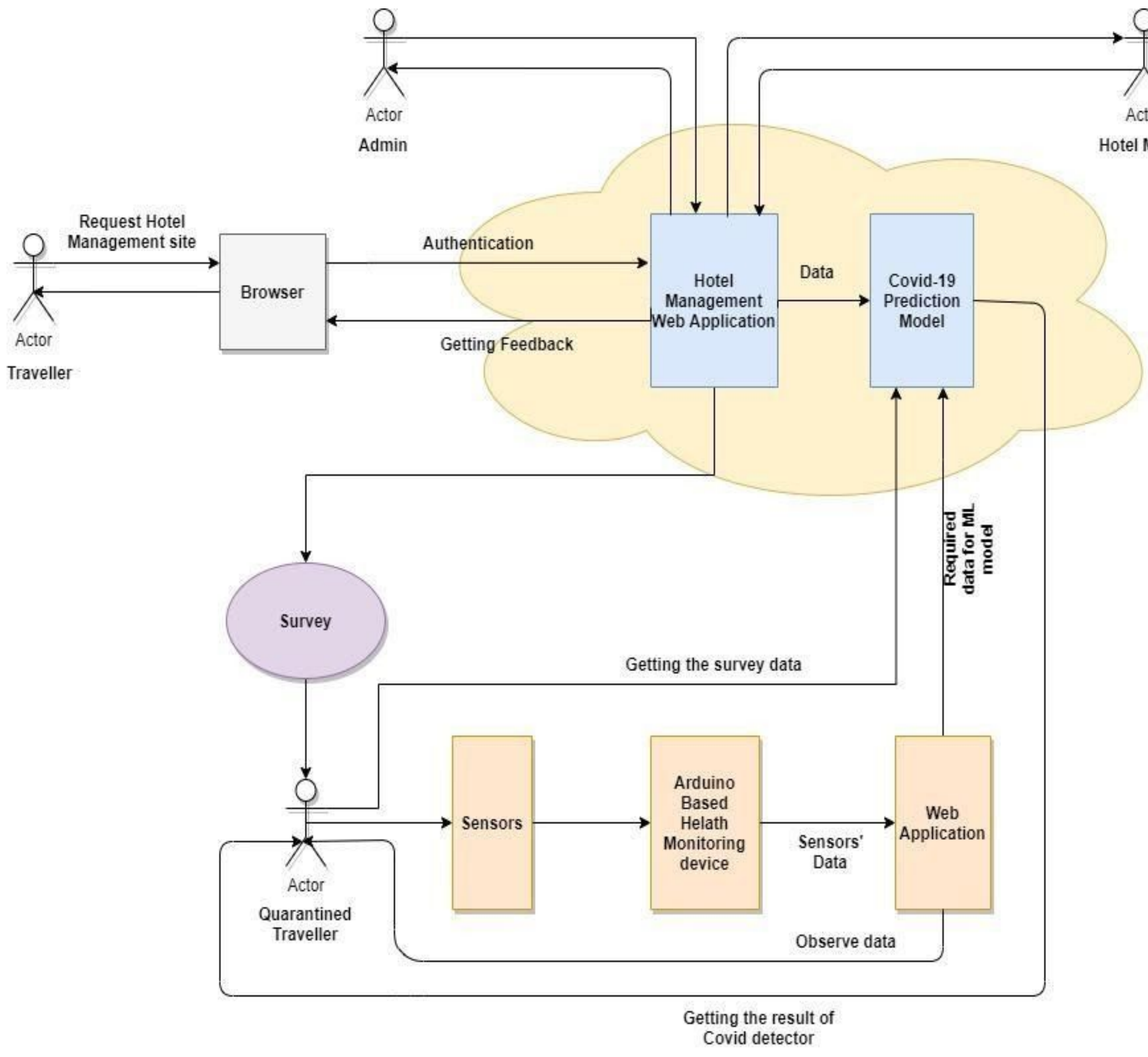
Here the main objective is to design a Remote Patient Health Monitoring System to diagnose the health condition of the COVID-19 patients. Giving care and health assistance to all the corona patients at critical stages with advanced medical facilities have become one of the major problems in the recent time. In hospitals where many patients whose physical conditions must be monitored frequently as a part of a diagnostic procedure, the need for a cost-effective and fast responding alert mechanism is inevitable. Proper implementation of such systems can provide timely warnings to the medical staff and doctors and their service can be activated in case of medical emergencies. Present-day systems use sensors that are hardwired to a PC next to the bed. The use of sensors detects the conditions of the patient and the data is collected and transferred using a microcontroller. Doctors and nurses need to visit the patient frequently to examine his/her current condition. In addition to this, use of multiple microcontrollers based intelligent systems provides high-level applicability in hospitals where many patients must be frequently monitored. For this, here we use the idea of network technology with wireless applicability, providing each patient a unique ID by which the doctor can easily identify the patient and his/her status of health parameters. Using the proposed system, data can be sent wirelessly to the Patient Monitoring System, allowing continuous monitoring of the patient. Contributing accuracy in measurements and providing security in proper alert mechanisms give this system a higher level of customer satisfaction and low-cost implementation in hospitals. Thus, the patient can engage in his daily activities in a comfortable atmosphere where distractions of hardwired sensors are not present. Physiological monitoring hardware can be easily implemented using simple interfaces of the sensors with a Microcontroller and can effectively be used for healthcare monitoring. This will allow development of such low-cost devices based on natural human-computer interfaces. The system we proposed here is efficient in monitoring the different physical parameters of many number bedridden patients and then in alerting the concerned medical authorities if these parameters bounce above its predefined critical values. Thus, remote monitoring and control refer to a field of industrial automation that is entering a new era with the development of wireless sensing devices. The Internet of Things (IoT) platform offers a promising technology to achieve the healthcare services, and can further improve the medical service systems. IoT wearable platforms can be used to collect the needed information of the user and its ambient environment and communicate such information wirelessly, where it is processed or stored for tracking the history of the user. Such connectivity with external devices and services will allow for taking preventive measures. Again, doctors can monitor large numbers of mild symptom patients without admitting to hospital and the infection exposure rate of Medical professionals will decrease by adopting this technology.

1.4.3 Objective of COVID-19 possibility detector

The infection rate of COVID-19 is increasing day by day. In Spite Of taking all necessary precautions, infection cannot be restricted. The number of hospitals and health workers are not sufficient to face this pandemic challenge. The availability of test kits is also not sufficient. The probability of doctors getting infected is also high as they are working in close contact with infected patients.

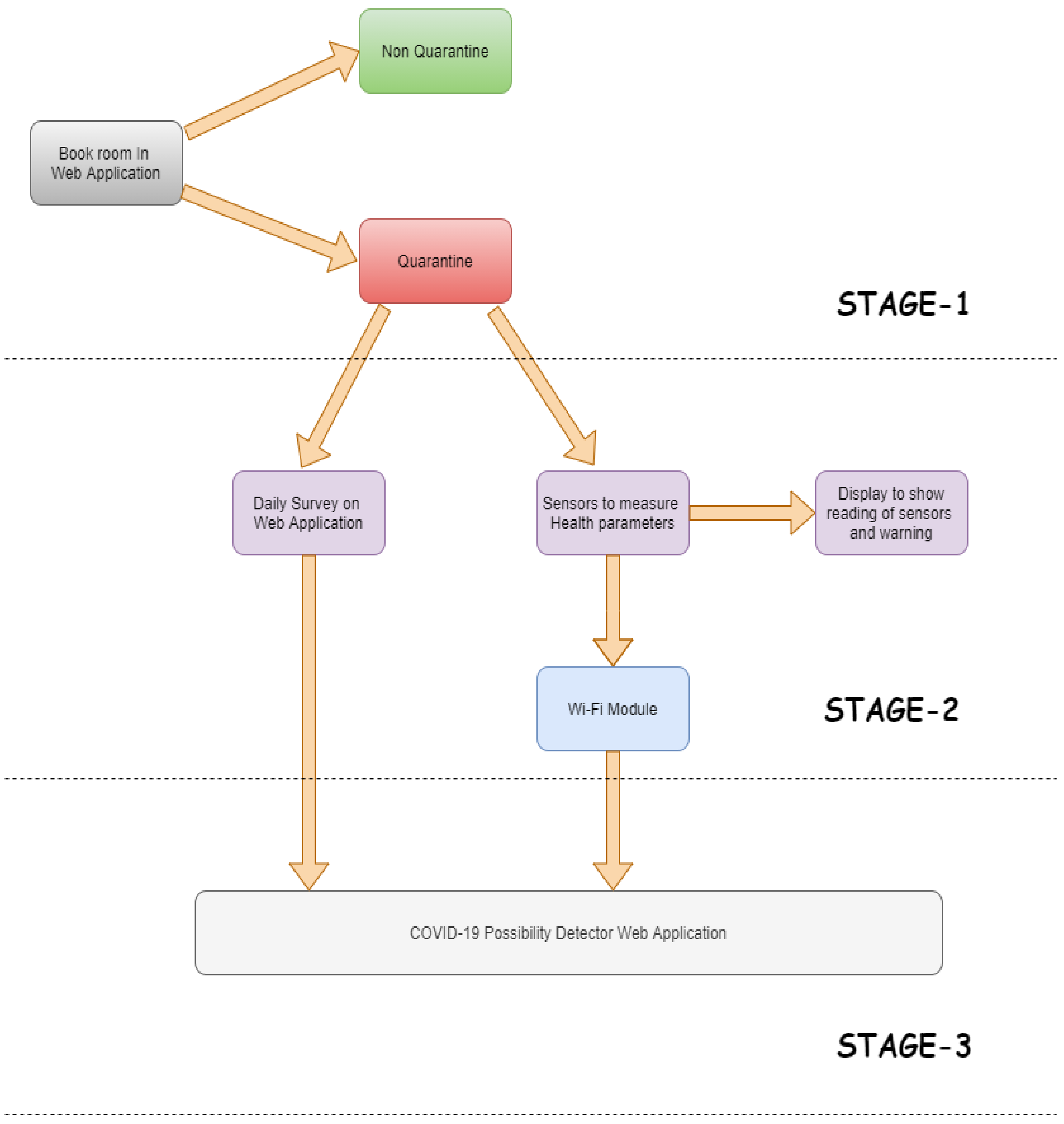
With this respect, the main objective of this project is to develop a detector using sensors and surveys through which millions of people can be tested with high accuracy and the probability of doctors to get infected is also slowed down as we are using contactless sensors and a small set of questions which can be asked using websites.

2. Diagram



Actor
Admin
Actor
Hotel Manager
Request Hotel Management site
Actor
Traveller
Browser
Authentication
Getting Feedback
Hotel Management Web Application
Data
Covid-19 Prediction Model
Required data for ML model
Survey
Getting the survey data
Actor
Quarantined Traveller
Sensors
Arduino Based Helath Monitoring device
Sensors' Data
Web Application
Observe data
Getting the result of Covid detector

3. Work Flow


Book room In Web Application
Non Quarantine
Quarantine
STAGE-1
Daily Survey on Web Application
Sensors to measure Health parameters
Display to show reading of sensors and warning
Wi-Fi Module
STAGE-2
COVID-19 Possibility Detector Web Application
STAGE-3

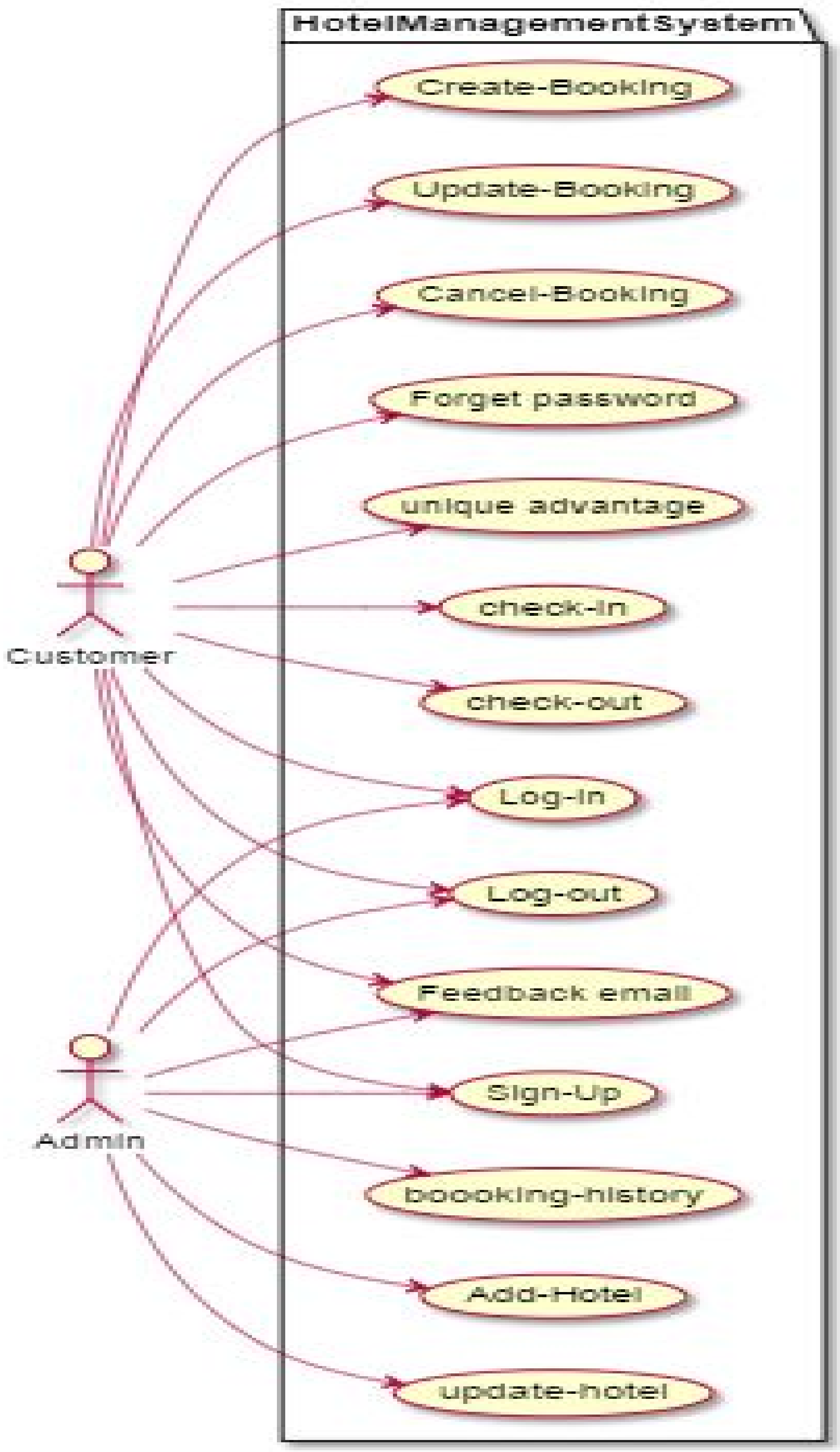
4. Workflow (STAGE -1)

4.1 Use Case Diagram

- List of Users
    - Customer
    - Admin
- Use-Cases
    - Customer can Log-in
    - Admin can log-in
    - Customer can Sign Up
    - Admin can Sign Up
    - Customer can Log out
    - Admin can Log-out
    - Customer can make a booking
    - Customer can cancel a booking
    - Customer can update a booking
    - Admin can update hotels
    - Admin can add hotels.
    - Customer can use Forget Password facility
    - Frequent Customer has unique Advantage
    - Customer has feedback mail after check-out
    - Customer can check-in
- Customer can check-out
- Admin check feedback mail

~~~
```plantuml
@startuml
left to right direction
actor "Customer" as Customer
actor "Admin" as Admin
package HotelManagementSystem {
  usecase "Sign-Up" as Cust_AD_UC1
  usecase "Log-In" as Cust_AD_UC2
  usecase "Log-out" as Cust_AD_UC3
  usecase "Feedback email" as Cust_AD_UC4
usecase "Create-Booking" as Cust_UC5
  usecase "Update-Booking" as Cust_UC6
  usecase "Cancel-Booking" as Cust_UC7
usecase "Forget password" as Cust_UC8
usecase "unique advantage" as Cust_UC9
usecase "check-in" as Cust_UC10
usecase "check-out" as Cust_UC11
usecase "Add-Hotel" as AD_UC1
usecase "update-hotel" as AD_UC2
usecase "boooking-history" as AD_UC3
~~~

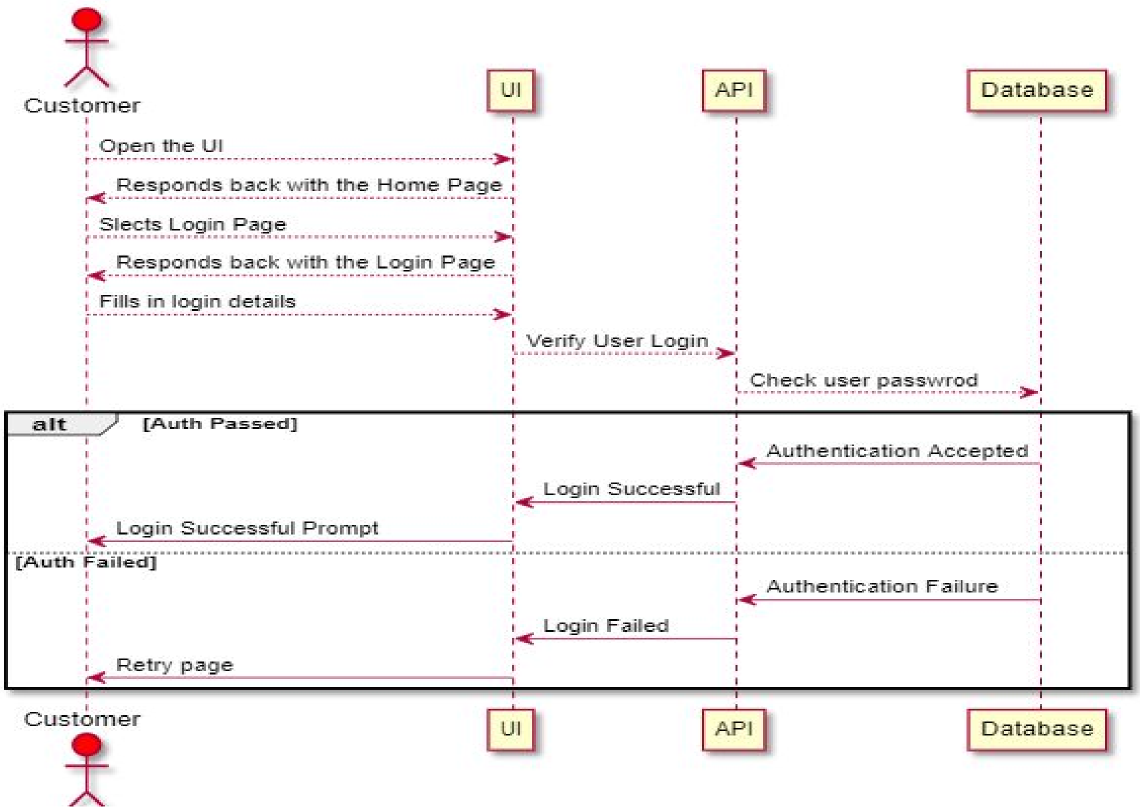
```
}
Customer --> Cust_AD_UC1
Customer --> Cust_AD_UC2
Customer --> Cust_AD_UC3
Customer --> Cust_AD_UC4
Customer --> Cust_UC5
Customer --> Cust_UC6
Customer --> Cust_UC7
Customer --> Cust_UC8
Customer --> Cust_UC9
Customer --> Cust_UC10
Customer --> Cust_UC11
Admin    --> Cust_AD_UC1
Admin   --> Cust_AD_UC2
Admin   -->  Cust_AD_UC3
Admin   --> Cust_AD_UC4
Admin   --> AD_UC1
Admin --> AD_UC2
Admin -->AD_UC3

@enduml
```
```
```


HotelManagementSystem
Create-Booking
Update-Booking
Cancel-Booking
Forget password
unique advantage
check-in
check-out
Log-in
Log-out
Feedback email
Sign-Up
boooking-history
Add-Hotel
update-hotel
Customer
Admin

4.2 Sequence Diagram

4.2.1 Sequence Diagram for Log-In for a Customer User

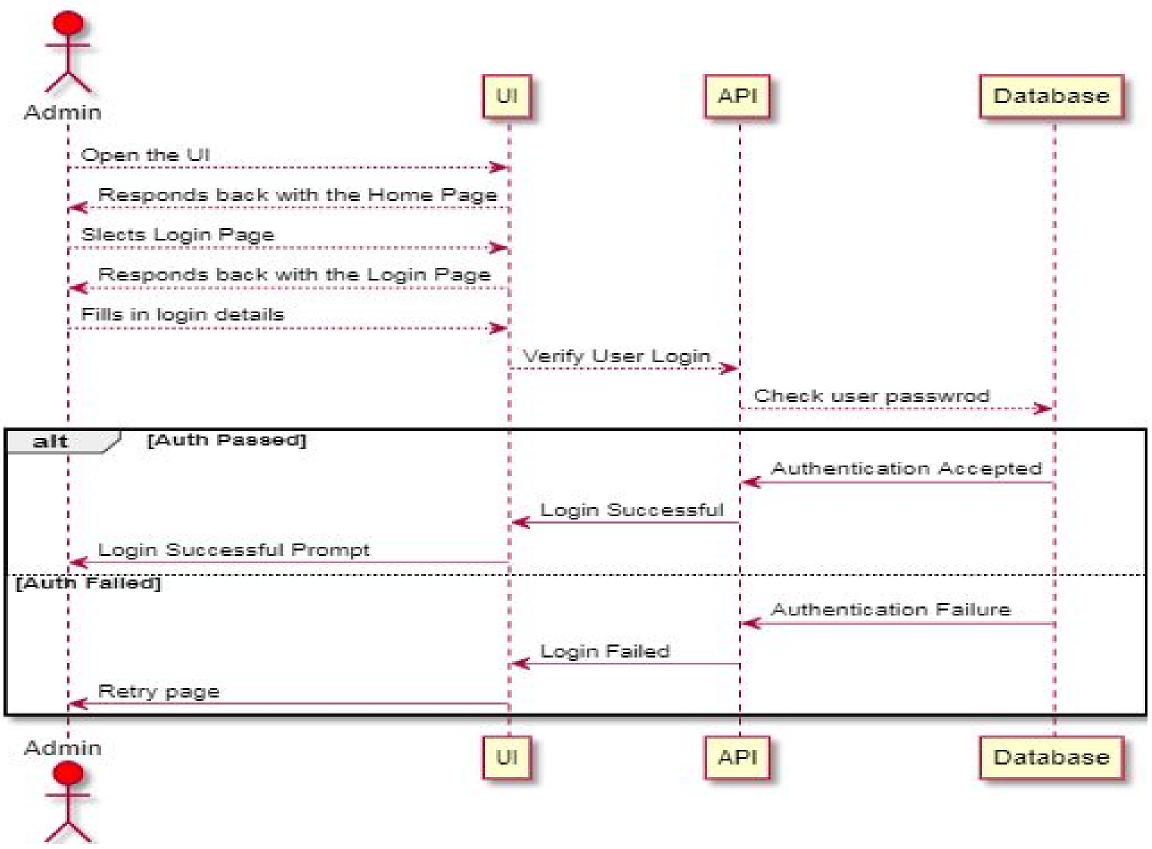
~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Login Page
UI --> Customer : Responds back with the Login Page
Customer --> UI : Fills in login details
UI --> API : Verify User Login
API --> Database : Check user passwrod
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Login Successful
    UI -> Customer : Login Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Login Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~



4.2.2 Sequence Diagram for Log-In for an Admin user

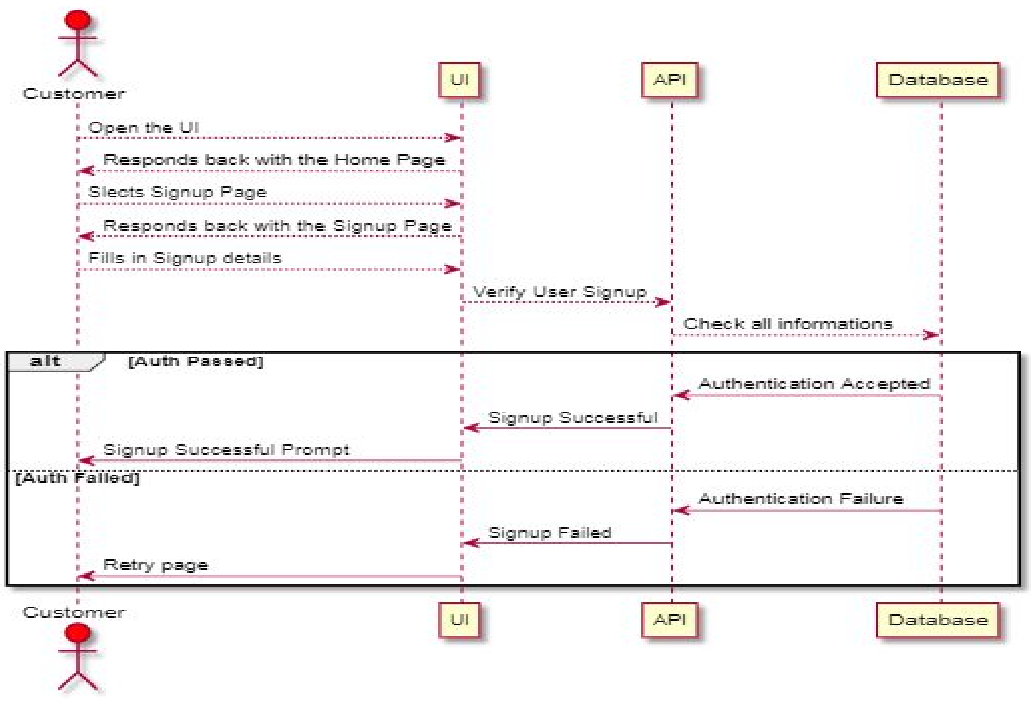
~~~
```plantuml
@startuml
actor Admin #red
~~~

~~~
Admin --> UI : Open the UI
UI --> Admin : Responds back with the Home Page
Admin --> UI : Slects Login Page
UI --> Admin : Responds back with the Login Page
Admin --> UI : Fills in login details
UI --> API : Verify User Login
API --> Database : Check user passwrod
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Login Successful
    UI -> Admin : Login Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Login Failed
    UI -> Admin: Retry page

end

@enduml
```
~~~

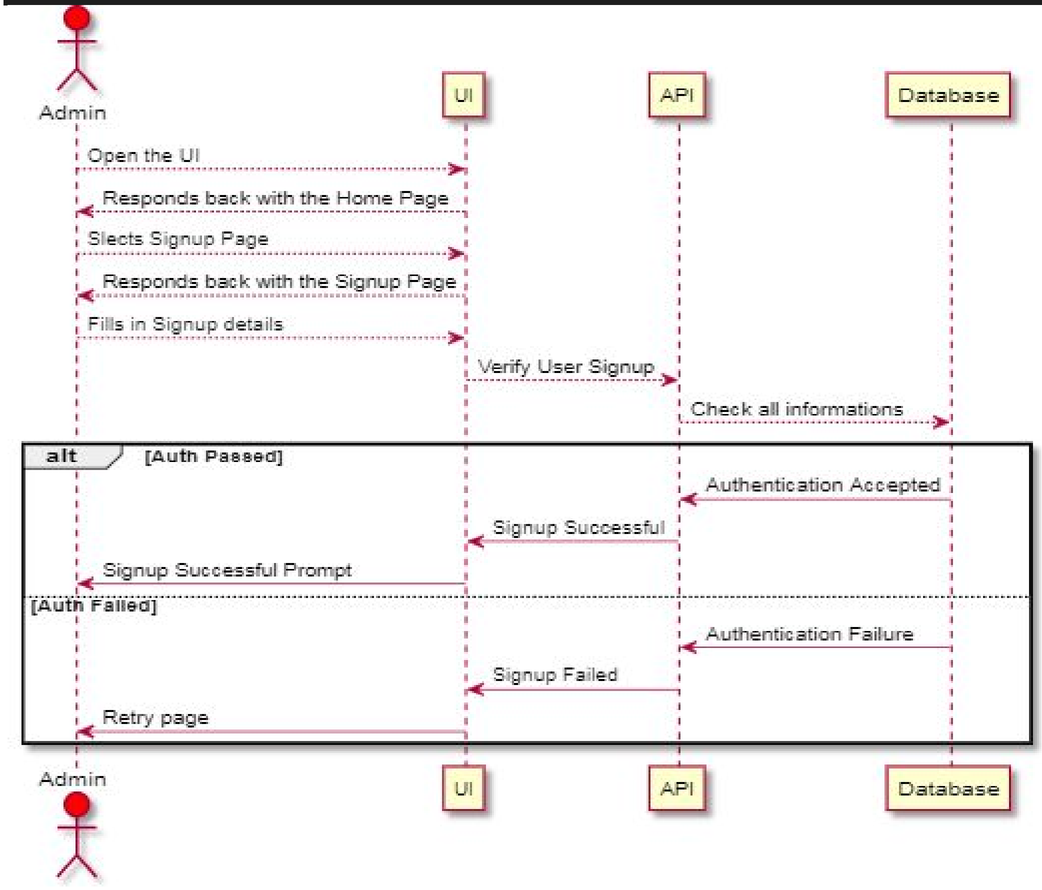


4.2.3 Sequence Diagram for Sign-up for a Customer User

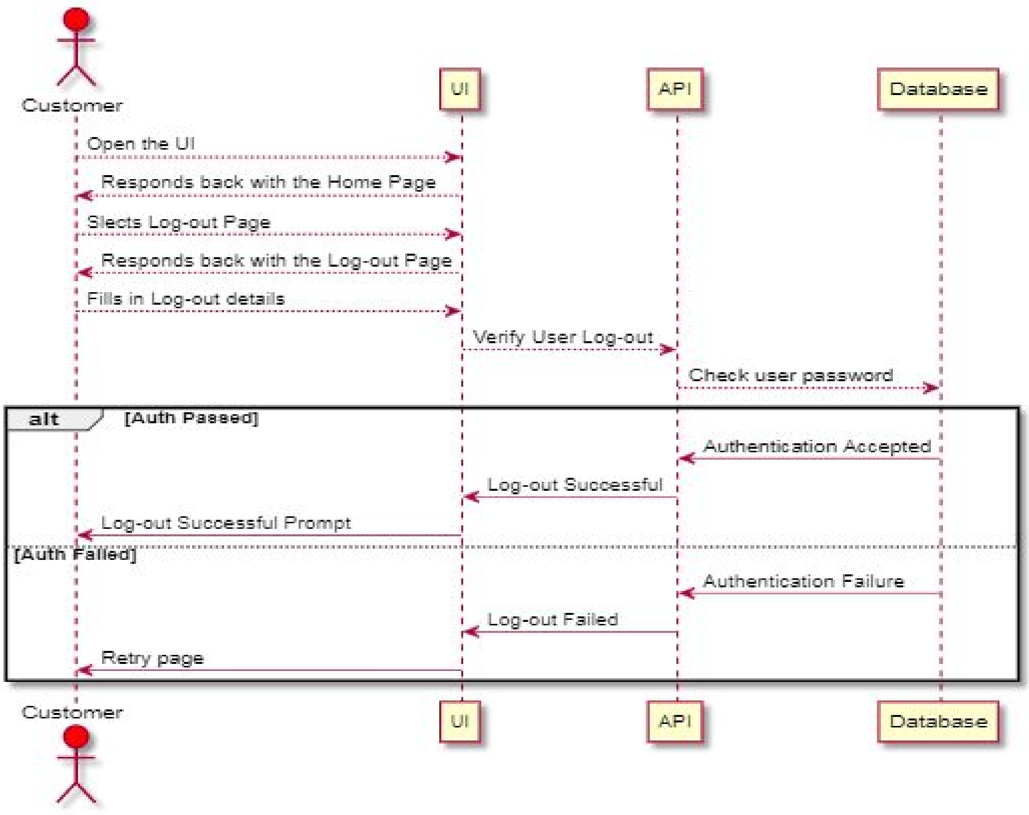
~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Signup Page
UI --> Customer : Responds back with the Signup Page
Customer --> UI : Fills in Signup details
~~~

```
UI --> API : Verify User Signup
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Signup Successful
    UI -> Customer : Signup Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Signup Failed
    UI -> Customer : Retry page

end

@enduml
```
```


Customer
UI
API
Database
Open the UI
Responds back with the Home Page
Slects Signup Page
Responds back with the Signup Page
Fills in Signup details
Verify User Signup
Check all informations
alt
[Auth Passed]
Authentication Accepted
Signup Successful
Signup Successful Prompt
[Auth Failed]
Authentication Failure
Signup Failed
Retry page
Customer
UI
API
Database

### *4.2.4 Sequence Diagram for Sign-up for an Admin User*

````
```plantuml
@startuml
actor Admin #red

Admin--> UI : Open the UI
UI --> Admin : Responds back with the Home Page
Admin--> UI : Slects Signup Page
UI --> Admin : Responds back with the Signup Page
Admin--> UI : Fills in Signup details
UI --> API : Verify User Signup
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Signup Successful
````

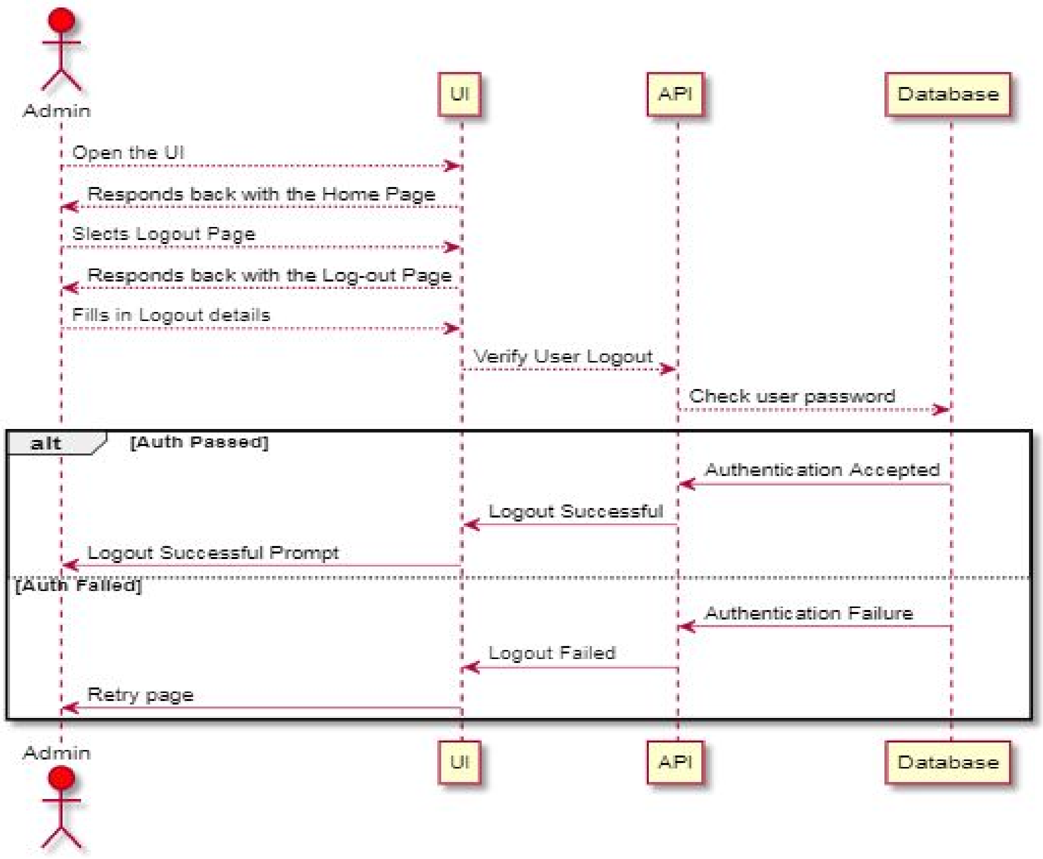
```
    UI -> Admin : Signup Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Signup Failed
    UI -> Admin : Retry page

end

@enduml
```



4.2.5 Sequence Diagram for Log-out for a Customer User

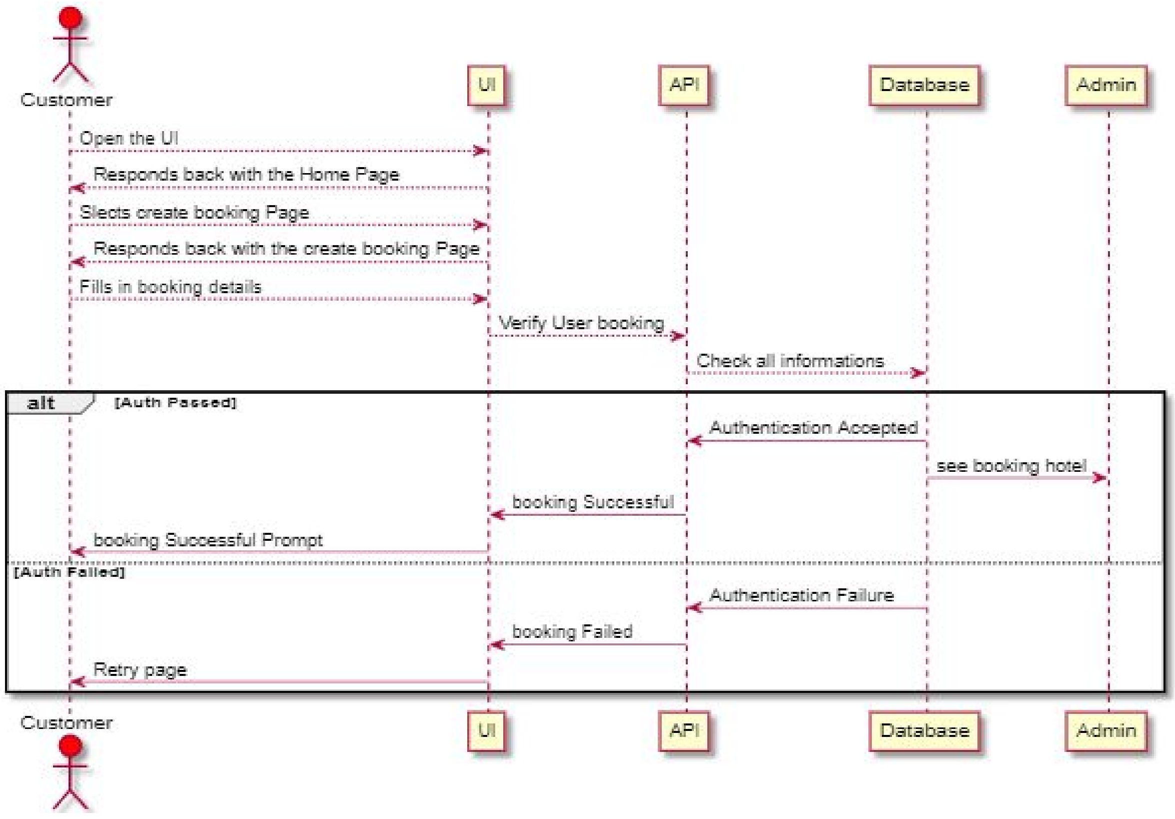
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Log-out Page
UI --> Customer : Responds back with the Log-out Page
Customer --> UI : Fills in Log-out details
UI --> API : Verify User Log-out
API --> Database : Check  user password
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Log-out Successful
    UI -> Customer : Log-out Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Log-out Failed
    UI -> Customer : Retry page

end

@enduml
```
```


Customer
UI
API
Database
Open the UI
Responds back with the Home Page
Slects Log-out Page
Responds back with the Log-out Page
Fills in Log-out details
Verify User Log-out
Check user password
alt
[Auth Passed]
Authentication Accepted
Log-out Successful
Log-out Successful Prompt
[Auth Failed]
Authentication Failure
Log-out Failed
Retry page
Customer
UI
API
Database

*4.2.6 Sequence Diagram for Log-out for an Admin User*

````
```plantuml
@startuml
actor Admin #red

Admin --> UI : Open the UI
UI --> Admin : Responds back with the Home Page
Admin --> UI : Slects Logout Page
UI --> Admin : Responds back with the Log-out Page
Admin --> UI : Fills in Logout details
UI --> API : Verify User Logout
API --> Database : Check user password
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Logout Successful
    UI -> Admin : Logout Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Logout Failed
    UI -> Admin : Retry page

end

@enduml
````

```
```



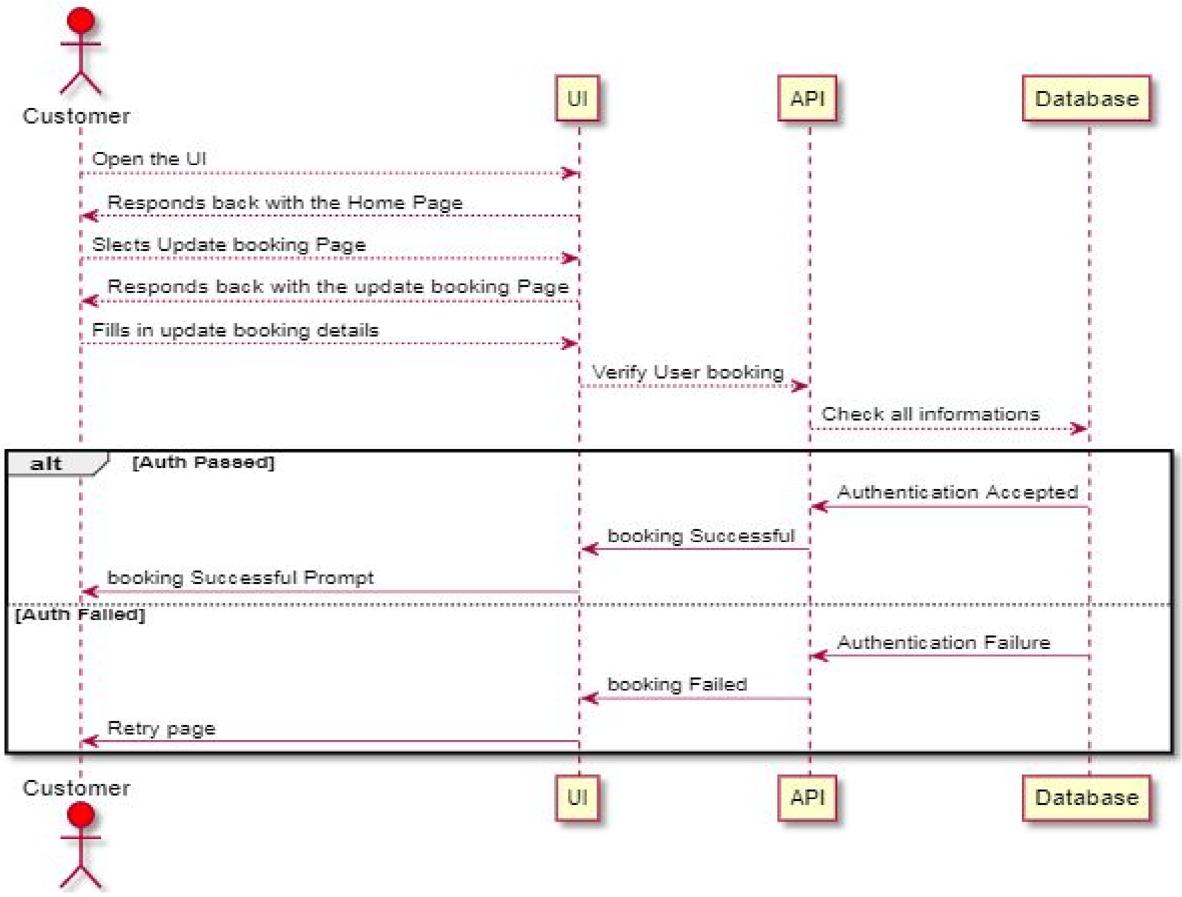
4.2.7 Sequence Diagram for create booking for a Customer User

```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects create booking Page
UI --> Customer : Responds back with the create booking  Page
Customer --> UI : Fills in booking details
UI --> API : Verify User booking
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    Database -> Admin: see booking hotel
    API -> UI : booking Successful
    UI -> Customer : booking Successful Prompt


else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : booking Failed
    UI -> Customer : Retry page

end

@enduml
```



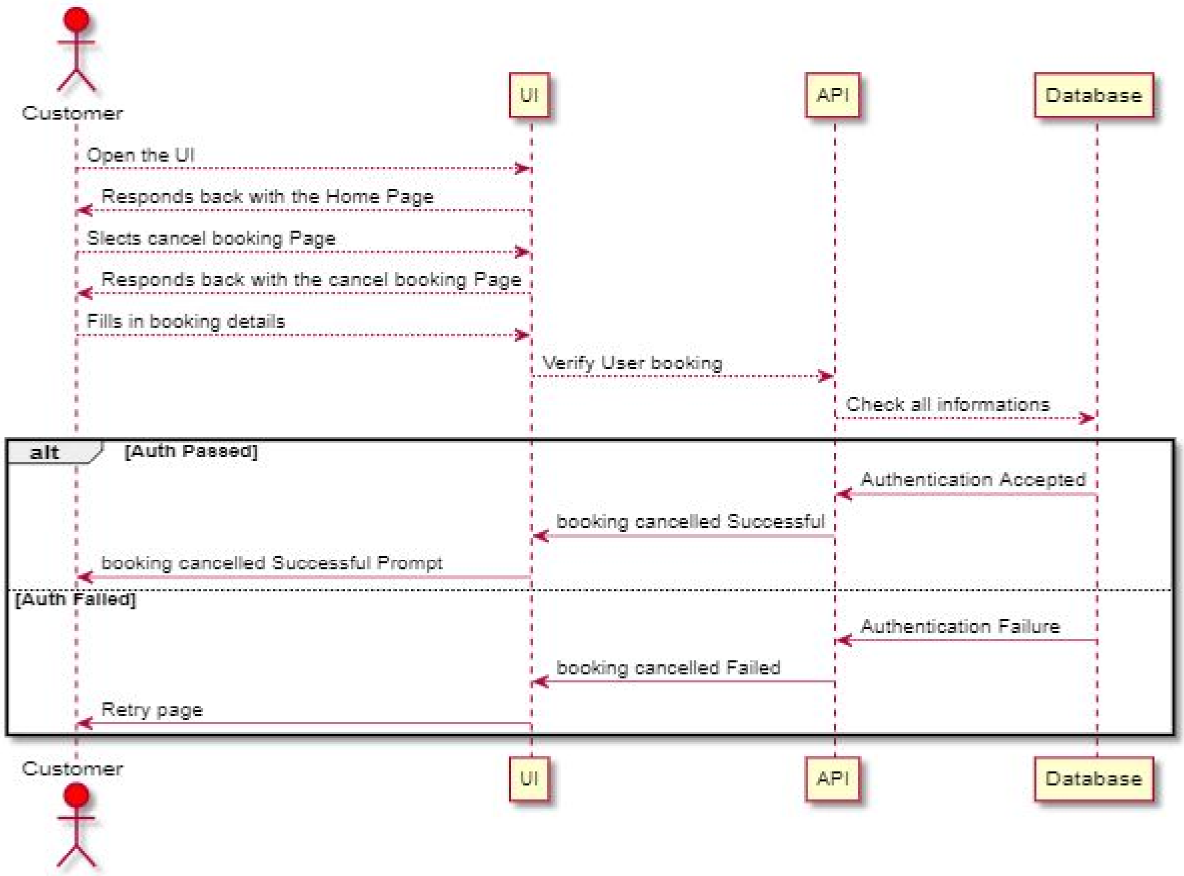
4.2.8 Sequence Diagram for Update booking for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Update booking Page
UI --> Customer : Responds back with the update  booking  Page
Customer --> UI : Fills in update booking details
UI --> API : Verify User booking
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : booking Successful
    UI -> Customer : booking Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : booking Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~



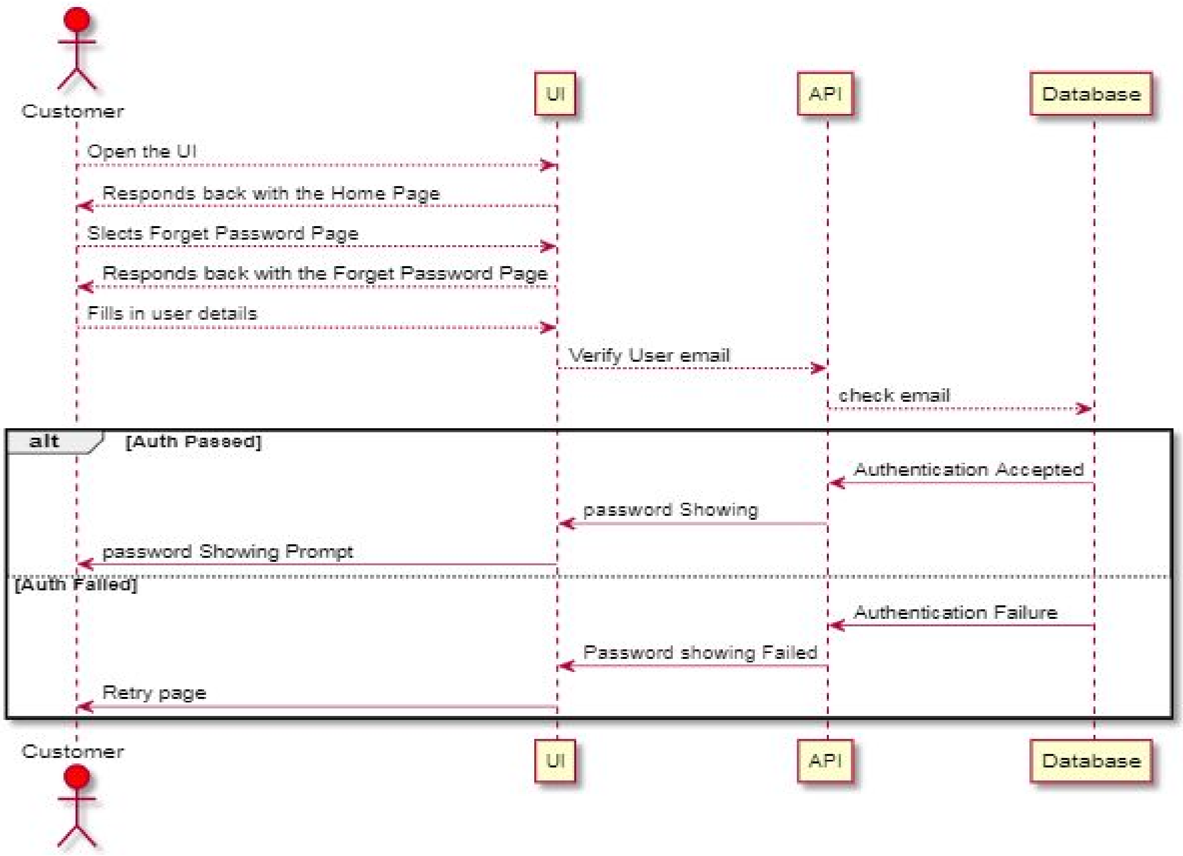
4.2.9 Sequence Diagram for cancel booking for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects cancel booking Page
UI --> Customer : Responds back with the cancel booking  Page
Customer --> UI : Fills in booking details
UI --> API : Verify User booking
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : booking cancelled Successful
    UI -> Customer : booking cancelled Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : booking cancelled Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~



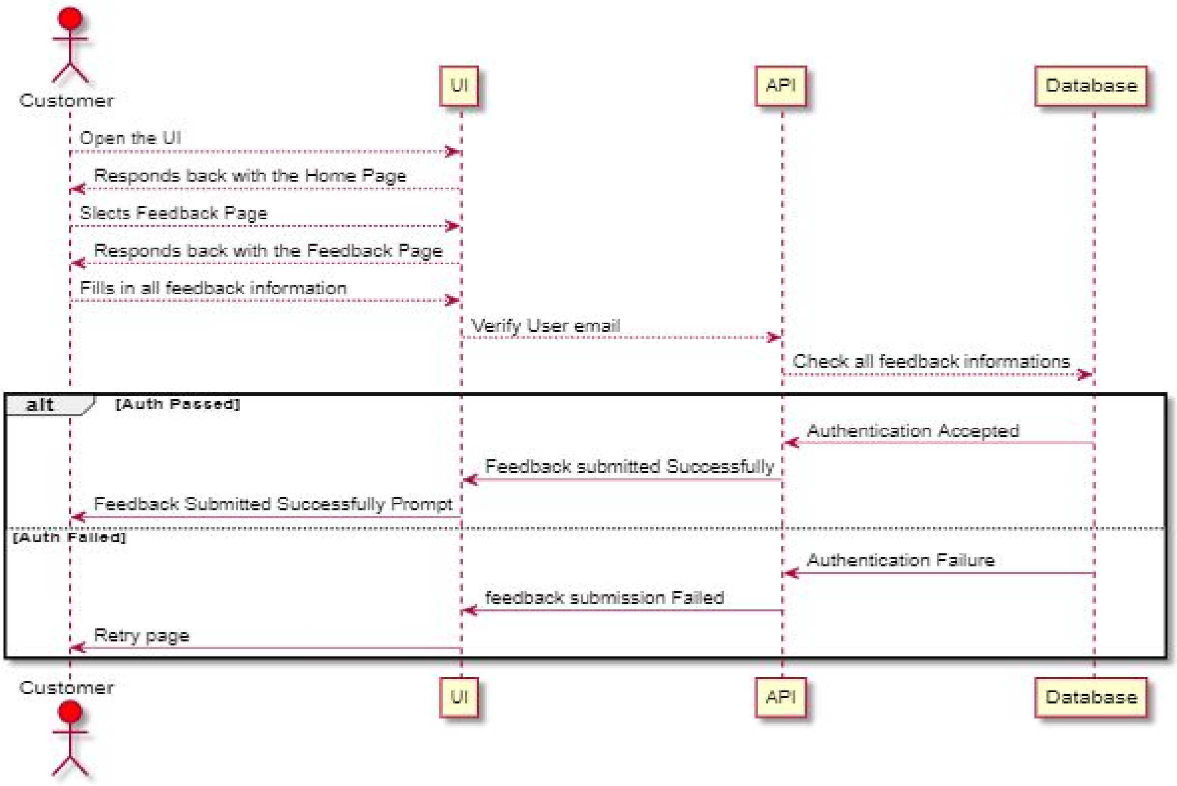
4.2.10 Sequence Diagram for Forget Password for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Forget Password Page
UI --> Customer : Responds back with the Forget Password  Page
Customer --> UI : Fills in user details
UI --> API : Verify User email
API --> Database : check email
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : password Showing
    UI -> Customer : password Showing Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Password showing Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~



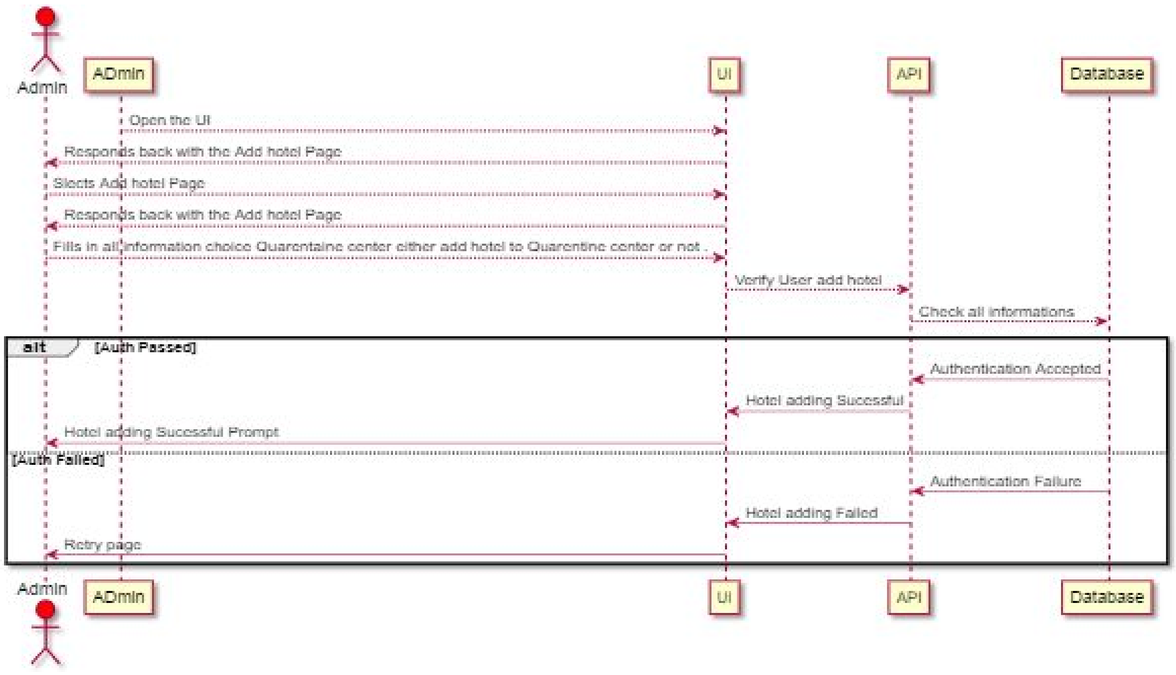
4.2.11 Sequence Diagram for Feedback email for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

Customer --> UI : Open the UI
UI --> Customer : Responds back with the Home Page
Customer --> UI : Slects Feedback Page
UI --> Customer : Responds back with the Feedback  Page
Customer --> UI : Fills in all feedback information
UI --> API : Verify User email
API --> Database : Check all feedback informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Feedback submitted Successfully
    UI -> Customer : Feedback Submitted Successfully Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI :feedback submission Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~


Customer
UI
API
Database
Open the UI
Responds back with the Home Page
Slects Feedback Page
Responds back with the Feedback Page
Fills in all feedback information
Verify User email
Check all feedback informations
alt
[Auth Passed]
Authentication Accepted
Feedback submitted Successfully
Feedback Submitted Successfully Prompt
[Auth Failed]
Authentication Failure
feedback submission Failed
Retry page
Customer
UI
API
Database

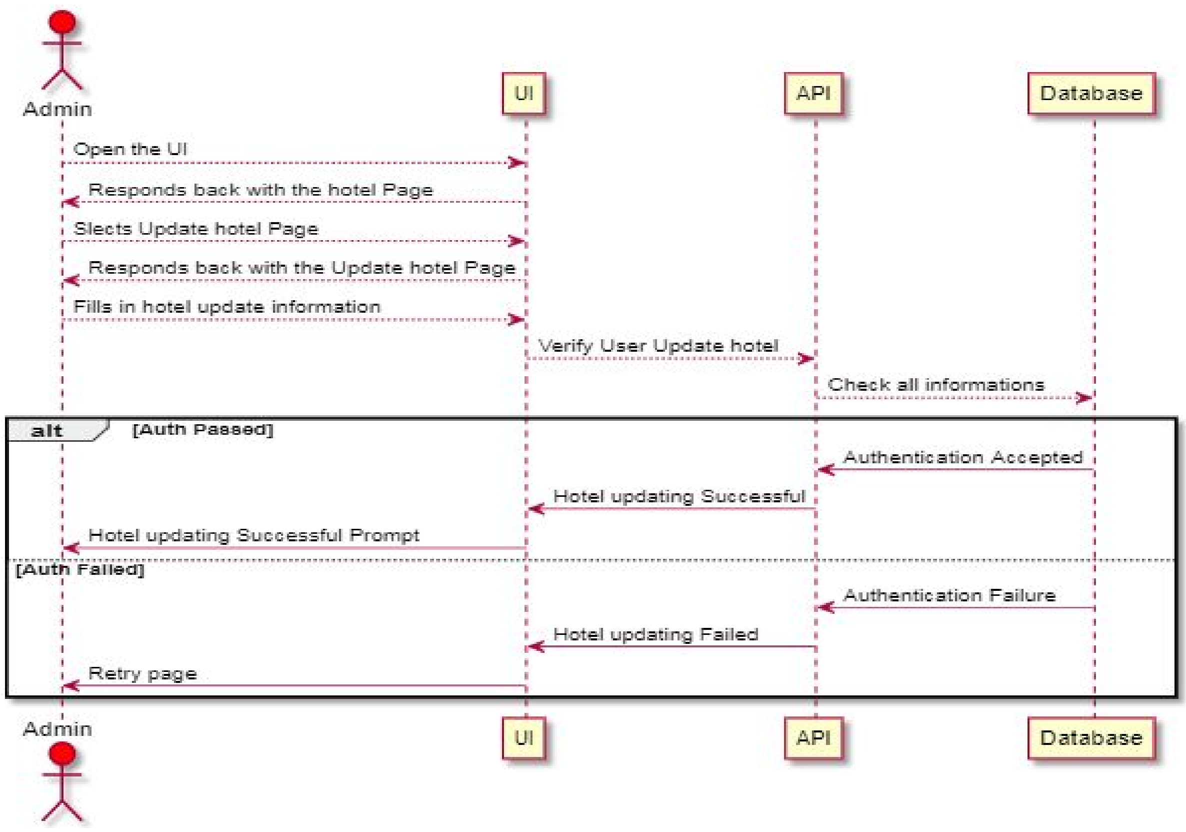
4.2.12 Sequence Diagram for Add hotel for an Admin User

~~~
```plantuml
@startuml
actor Admin #red

ADmin --> UI : Open the UI
UI --> Admin : Responds back with the Add hotel Page
Admin --> UI : Slects Add hotel Page
UI --> Admin: Responds back with the Add hotel Page
Admin--> UI : Fills in all information choice Quarentaine center either add hotel to Q
uarentine center or not .
UI --> API : Verify User add hotel
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Hotel adding Sucessful
    UI -> Admin :Hotel adding Sucessful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Hotel adding Failed
    UI -> Admin : Retry page

end

@enduml
```
~~~



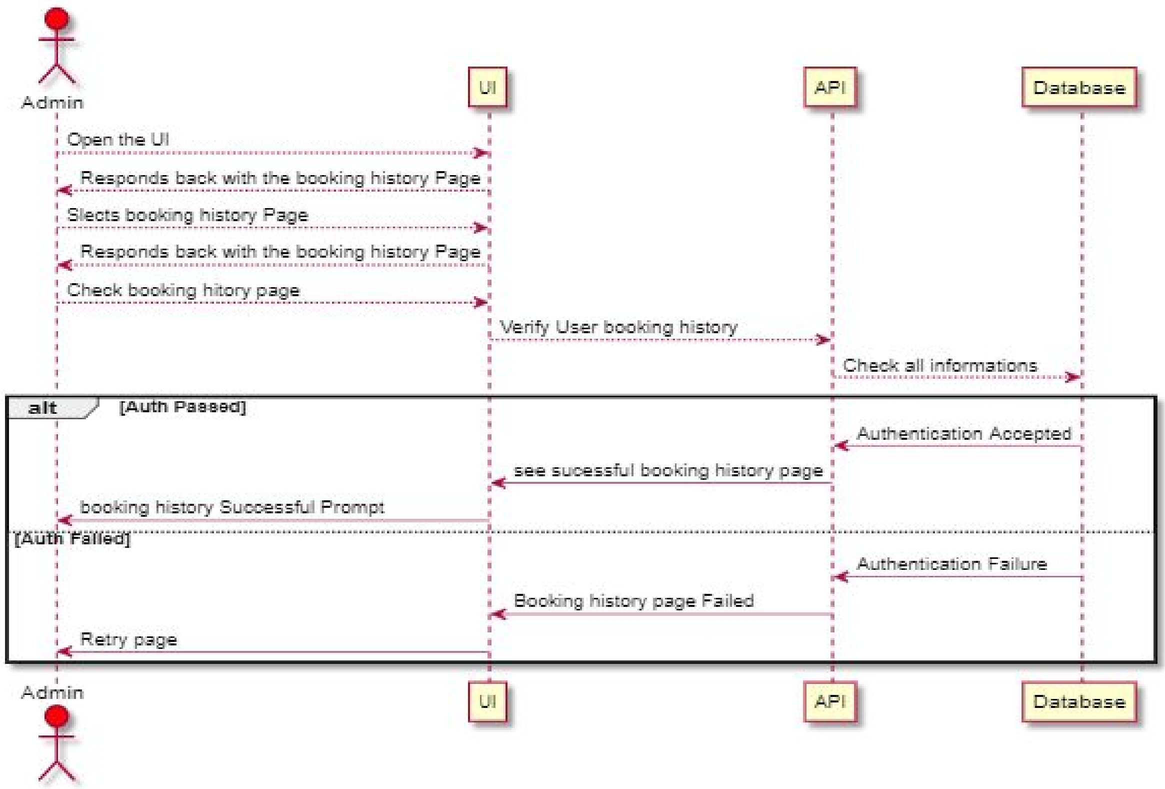
4.2.13 Sequence Diagram for update hotel for an Admin User

~~~
```plantuml
@startuml
actor Admin #red

Admin --> UI : Open the UI
UI --> Admin : Responds back with the hotel Page
Admin --> UI : Slects Update hotel Page
UI --> Admin : Responds back with the Update hotel  Page
Admin --> UI : Fills in hotel update information
UI --> API : Verify User Update hotel
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Hotel updating Successful
    UI -> Admin : Hotel updating Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Hotel updating Failed
    UI -> Admin : Retry page

end

@enduml
```
~~~



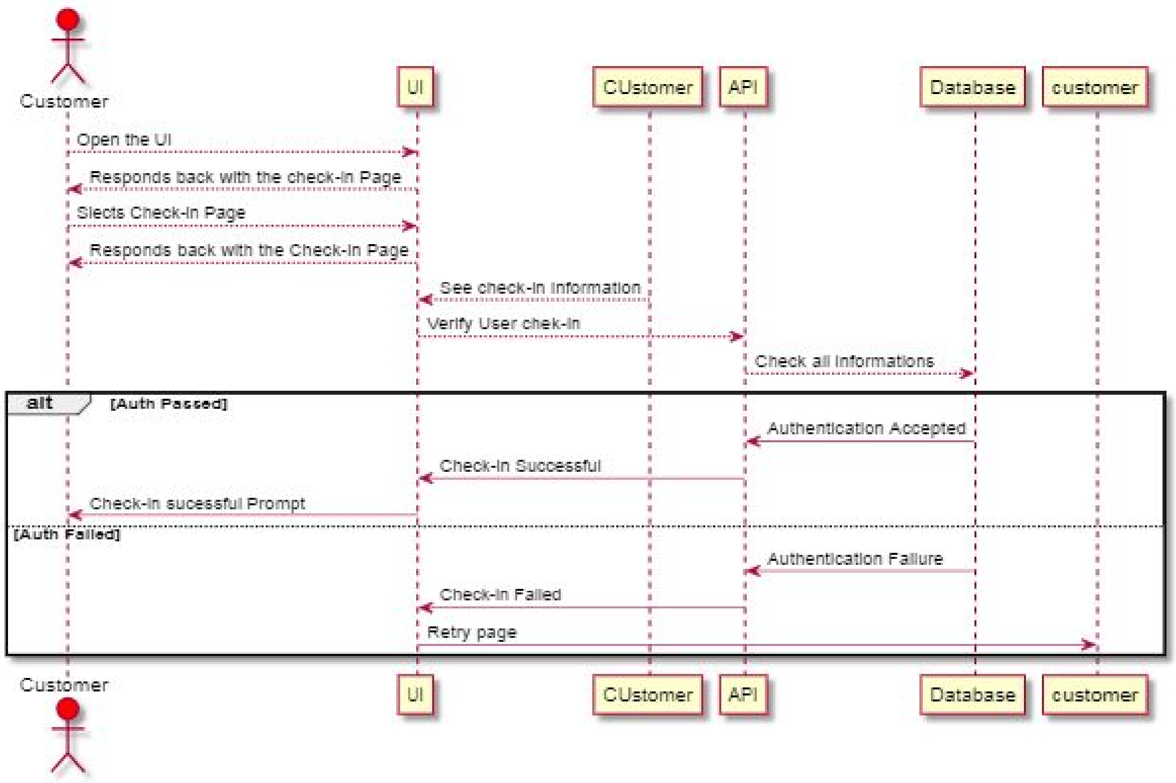
4.2.14 Sequence Diagram for booking history for an Admin User

~~~
```plantuml
@startuml
actor Admin #red

Admin --> UI : Open the UI
UI --> Admin : Responds back with the booking history Page
Admin --> UI : Slects booking history Page
UI --> Admin : Responds back with the booking history Page
Admin --> UI : Check booking hitory page
UI --> API : Verify User booking history
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : see sucessful booking history page
    UI -> Admin : booking history Successful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Booking history page Failed
    UI -> Admin : Retry page

end

@enduml
```
~~~


Admin
UI
API
Database
Open the UI
Responds back with the booking history Page
Slects booking history Page
Responds back with the booking history Page
Check booking hitory page
Verify User booking history
Check all informations
alt
[Auth Passed]
Authentication Accepted
see sucessful booking history page
booking history Successful Prompt
[Auth Failed]
Authentication Failure
Booking history page Failed
Retry page
Admin
UI
API
Database

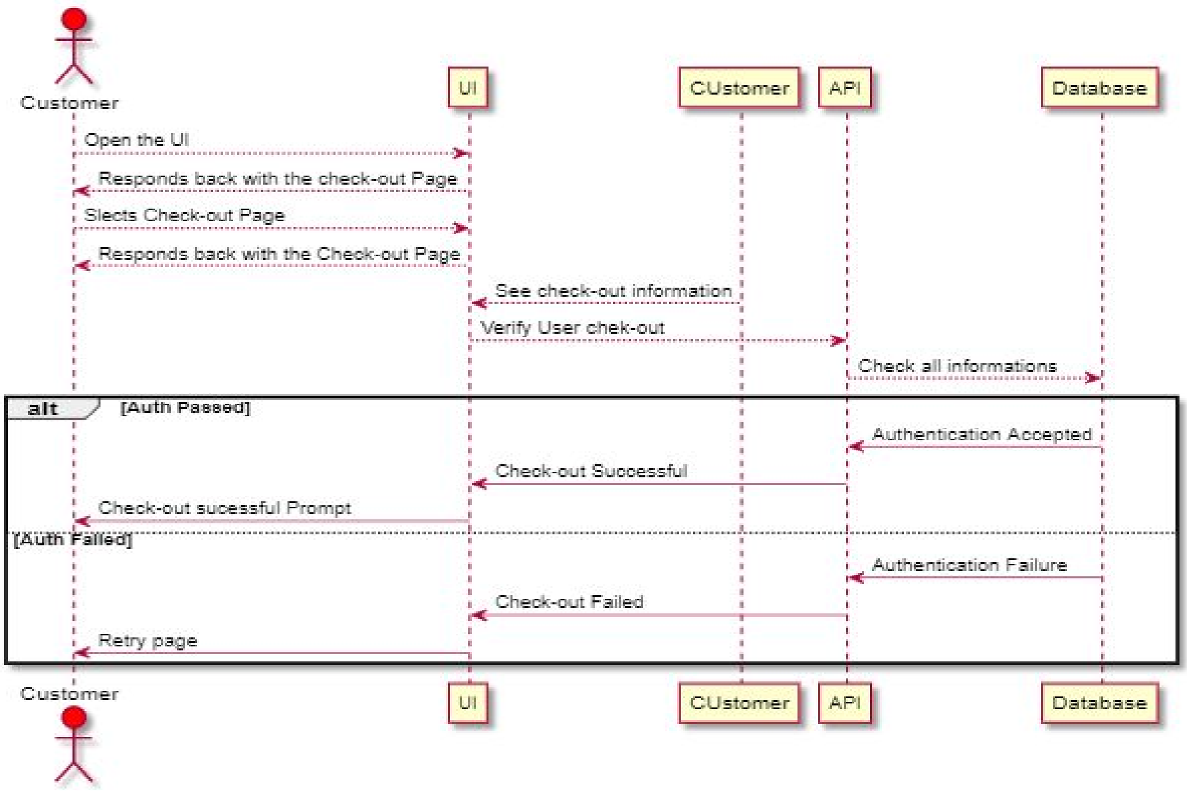
4.2.15 Sequence Diagram for check in for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

 Customer--> UI : Open the UI
UI --> Customer : Responds back with the check-in Page
Customer --> UI : Slects Check-in Page
UI --> Customer : Responds back with the Check-in  Page
CUstomer --> UI : See check-in information
UI --> API : Verify User chek-in
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Check-in Successful
    UI -> Customer : Check-in sucessful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Check-in Failed
    UI -> customer : Retry page

end

@enduml
```
~~~



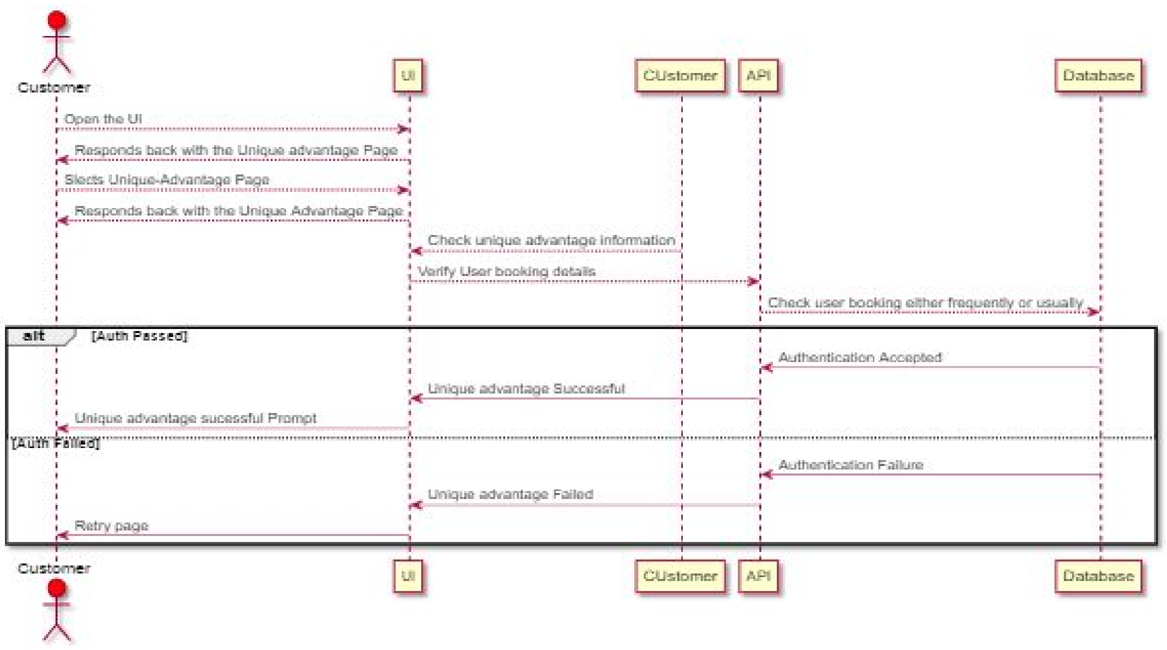
4.2.16 Sequence Diagram for check-out for a Customer User

~~~
```plantuml
@startuml
actor Customer #red

 Customer--> UI : Open the UI
UI --> Customer : Responds back with the check-out Page
Customer --> UI : Slects Check-out Page
UI --> Customer : Responds back with the Check-out Page
CUstomer --> UI : See check-out information
UI --> API : Verify User chek-out
API --> Database : Check all informations
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Check-out Successful
    UI -> Customer : Check-out sucessful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Check-out Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~



4.2.17 Sequence Diagram for unique advantage for a Customer User

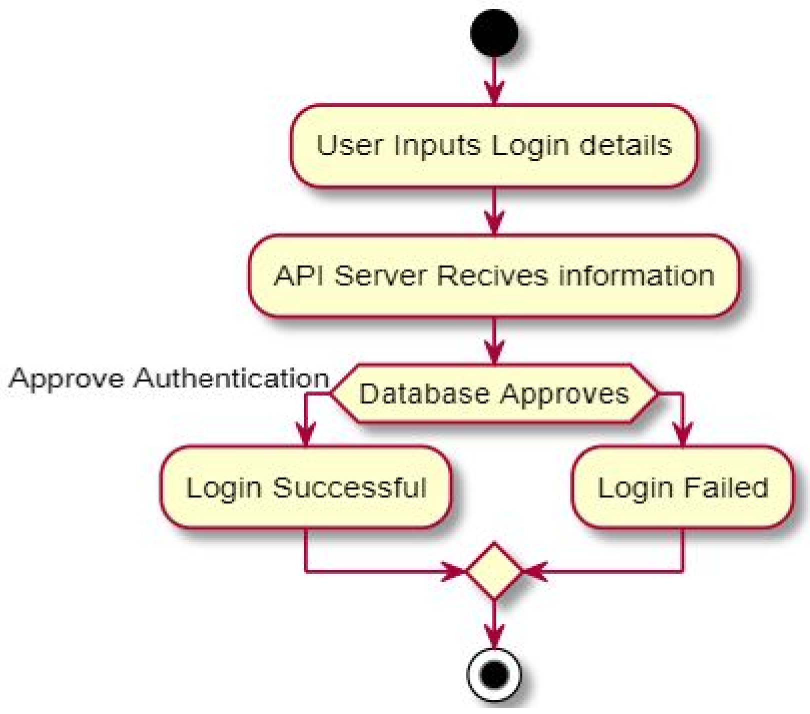
~~~
```plantuml
@startuml
actor Customer #red

 Customer--> UI : Open the UI
UI --> Customer : Responds back with the Unique advantage Page
Customer --> UI : Slects Unique-Advantage Page
UI --> Customer : Responds back with the Unique Advantage  Page
CUstomer --> UI : Check unique advantage information
UI --> API : Verify User booking details
API --> Database : Check user booking either frequently or usually
alt Auth Passed

    Database -> API: Authentication Accepted
    API -> UI : Unique advantage Successful
    UI -> Customer : Unique advantage sucessful Prompt

else Auth Failed

    Database -> API: Authentication Failure
    API -> UI : Unique advantage Failed
    UI -> Customer : Retry page

end

@enduml
```
~~~


Customer
UI
CUstomer
API
Database
Open the UI
Responds back with the Unique advantage Page
Slects Unique-Advantage Page
Responds back with the Unique Advantage Page
Check unique advantage information
Verify User booking details
Check user booking either frequently or usually
alt
[Auth Passed]
Authentication Accepted
Unique advantage Successful
Unique advantage sucessful Prompt
[Auth Failed]
Authentication Failure
Unique advantage Failed
Retry page

4.3 Activity Diagram

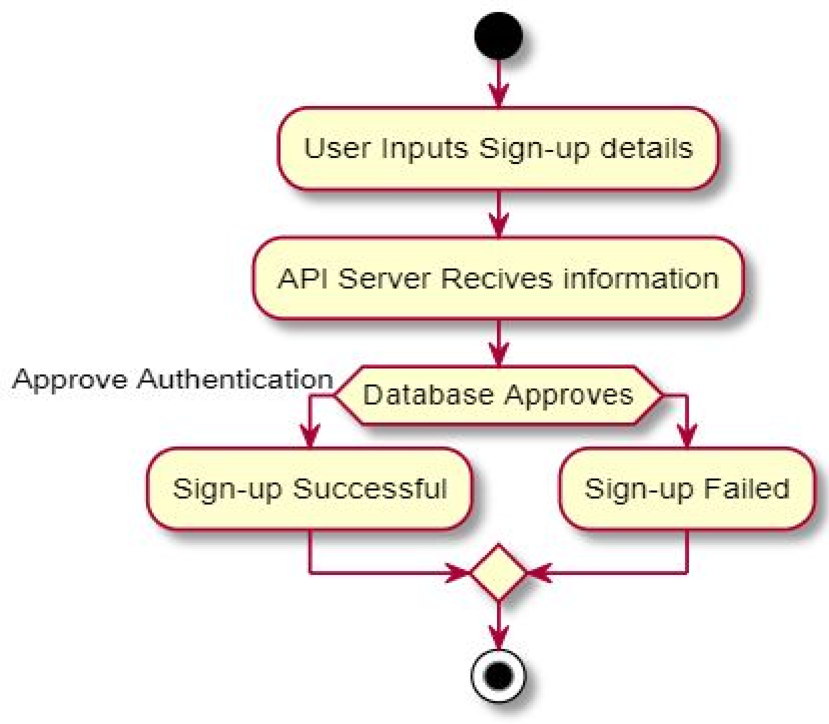
4.3.1 Activity Diagram for User Login

~~~
```plantuml
@startuml
start
:User Inputs Login details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Login Successful;
else
:Login Failed;
endif
stop

@enduml
```
~~~



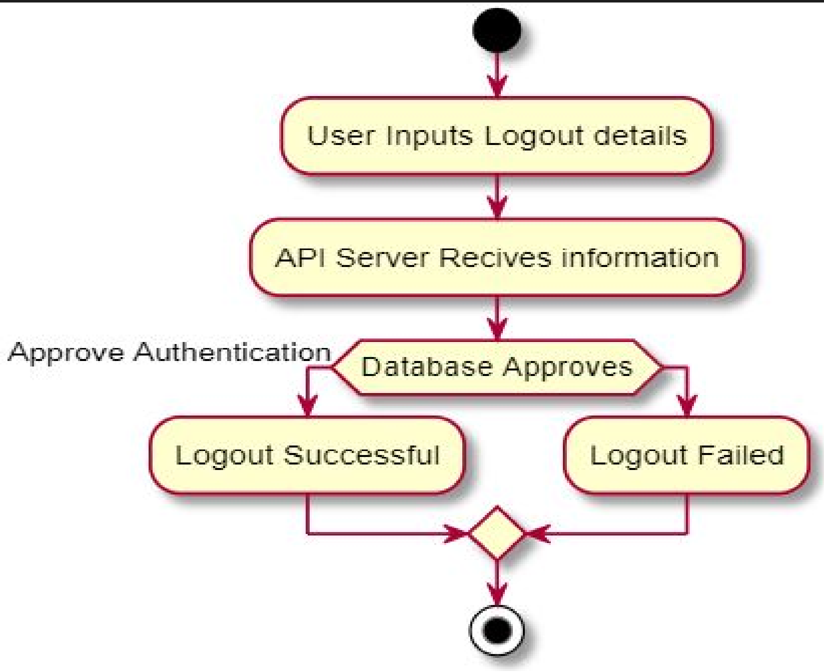
4.3.2 Activity Diagram for User sign-up

~~~
```plantuml
@startuml
start
:User Inputs Sign-up details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Sign-up Successful;
else
:Sign-up Failed;
endif
stop

@enduml
```
~~~



4.3.3 Activity Diagram for User Logout

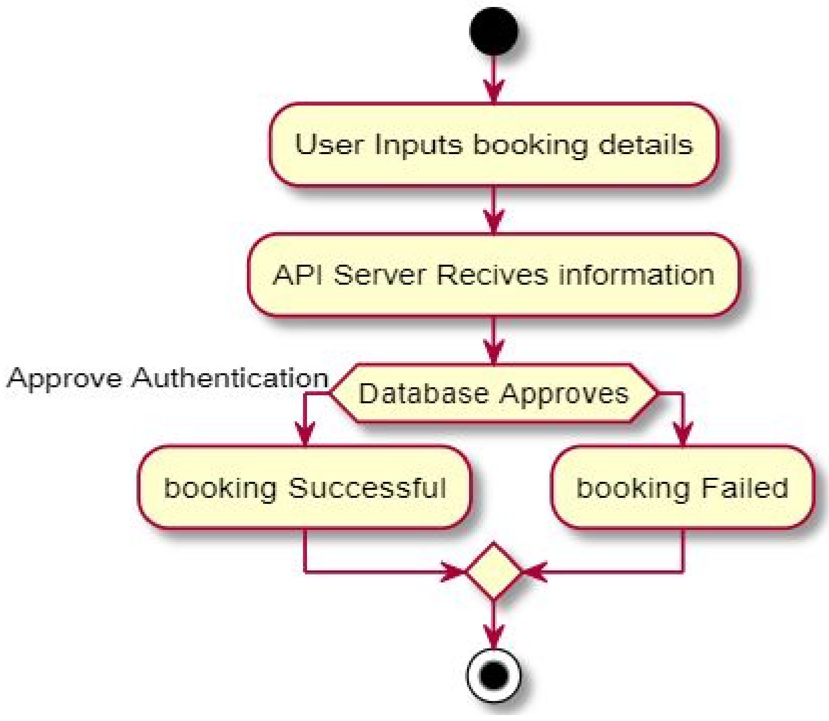
~~~
```plantuml
@startuml
start
:User Inputs Logout details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Logout Successful;
else
:Logout Failed;
endif
stop

@enduml
```
~~~



4.3.4 Activity Diagram for User create booking

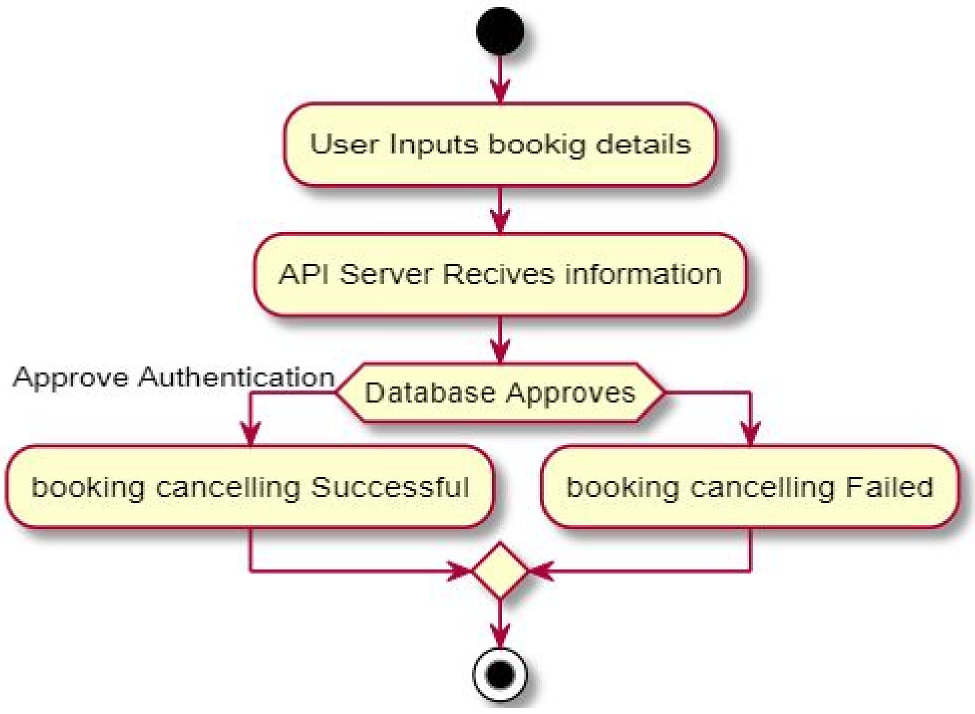
~~~
```plantuml
@startuml
~~~

```
start
:User Inputs booking details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :booking Successful;
else
:booking Failed;
endif
stop

@enduml
```



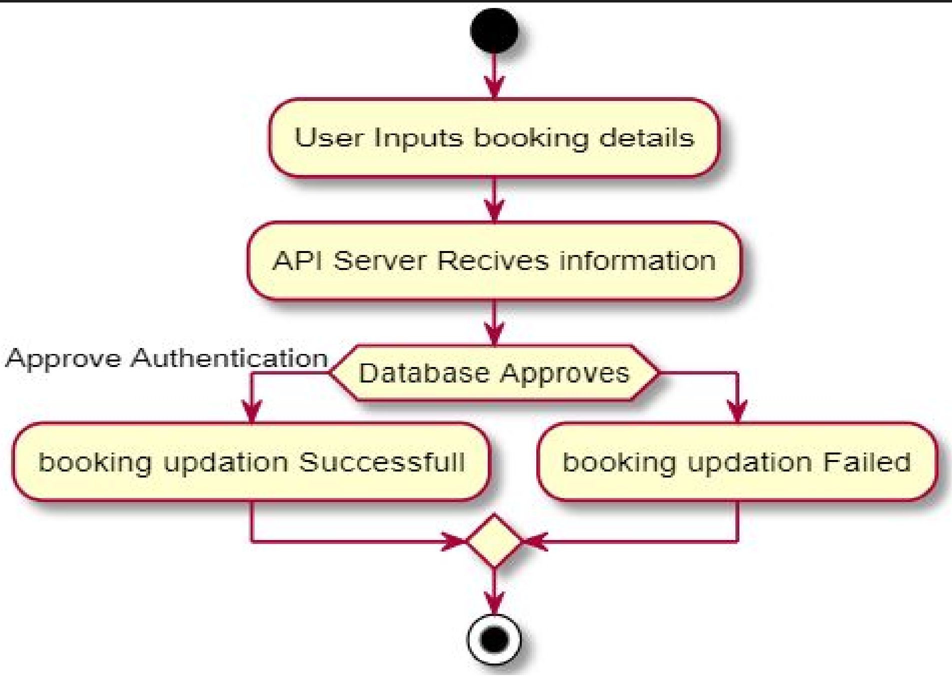
4.3.5 Activity Diagram for User cancel booking

~~~
```plantuml
@startuml
start
:User Inputs bookig details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :booking cancelling Successful;
else
:booking cancelling Failed;
endif
stop

@enduml
~~~


User Inputs bookig details
API Server Recives information
Approve Authentication
Database Approves
booking cancelling Successful
booking cancelling Failed

4.3.6 Activity Diagram for User update booking

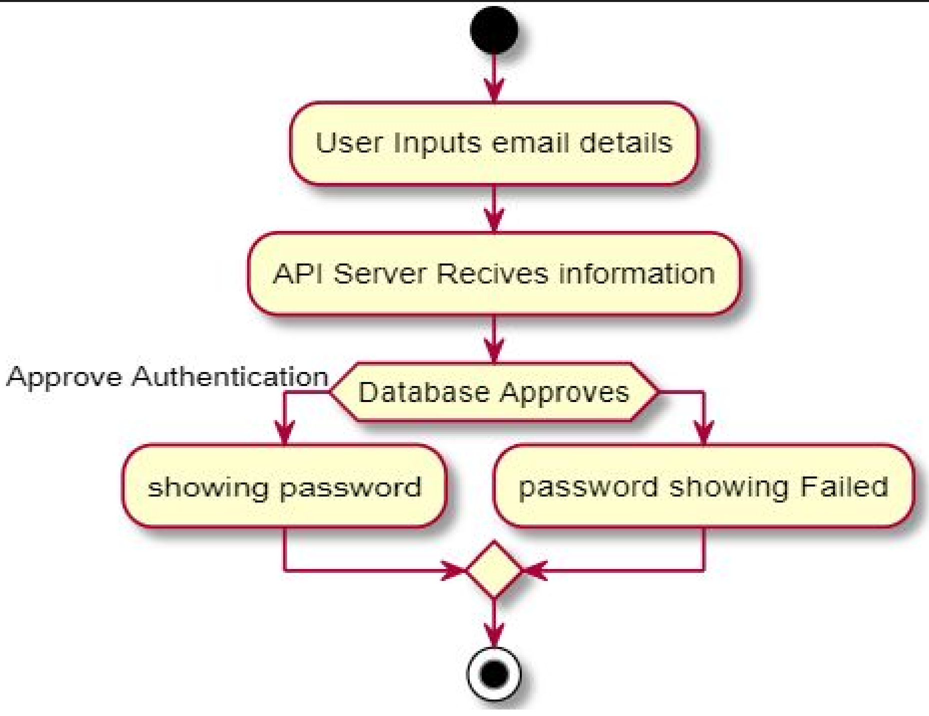
~~~
```plantuml
@startuml
start
:User Inputs booking details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :booking updation Successfull;
else
:booking updation Failed;
endif
stop

@enduml
```
~~~



4.3.7 Activity Diagram for User forget password

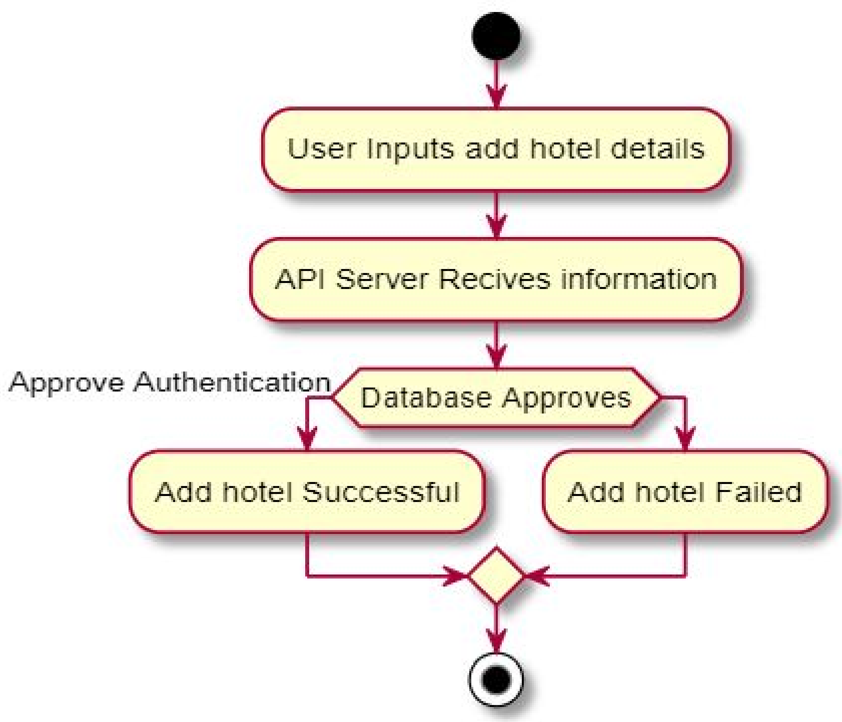
~~~
```plantuml
@startuml
~~~

```
start
:User Inputs email details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :showing password;
else
:password showing Failed;
endif
stop

@enduml
```



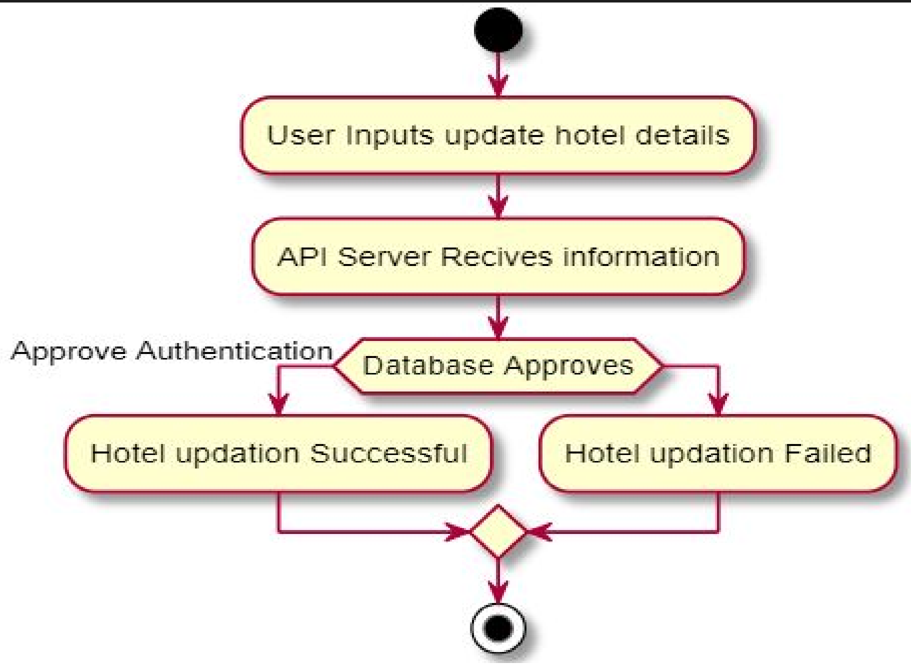
4.3.8 Activity Diagram for Add hotel

~~~
```plantuml
@startuml
start
:User Inputs add hotel details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Add hotel Successful;
else
:Add hotel Failed;
endif
stop

@enduml
```
~~~



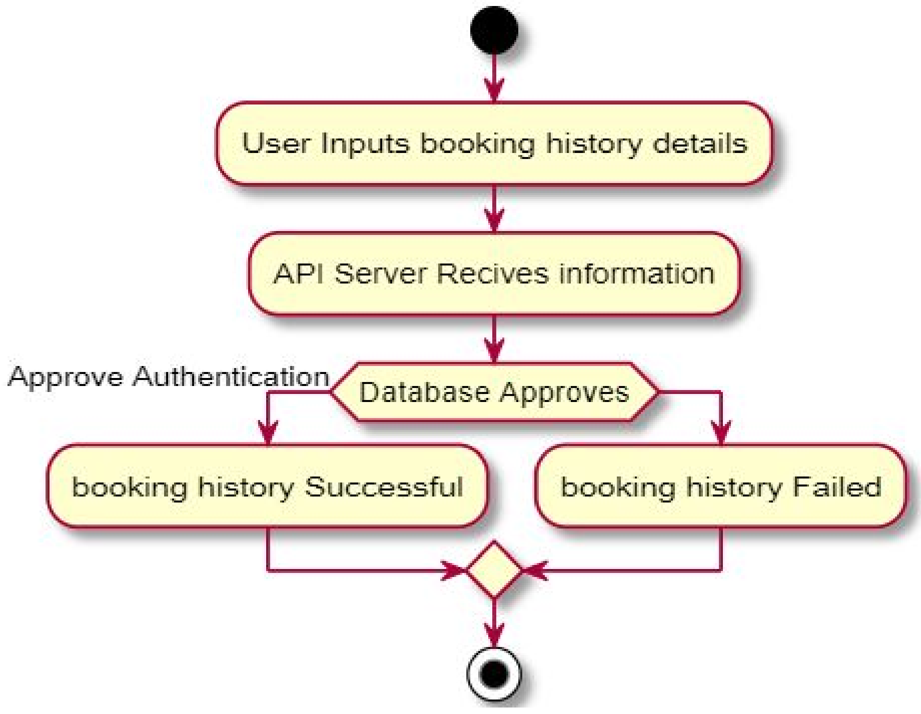
4.3.9 Activity Diagram for User update hotel

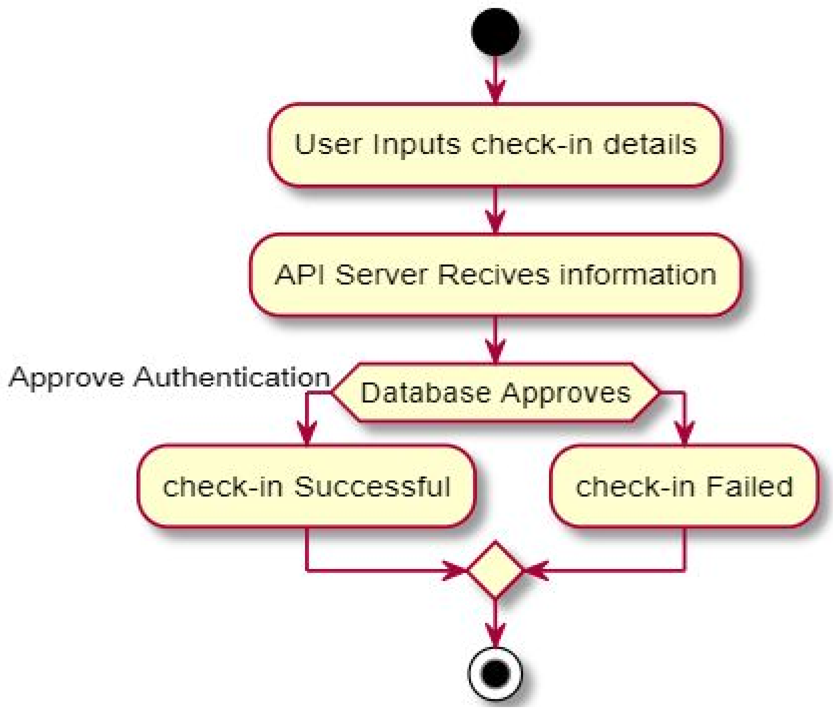
~~~
```plantuml
@startuml
start
:User Inputs update hotel details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Hotel updation Successful;
else
:Hotel updation Failed;
endif
stop

@enduml
```
~~~



4.3.10 Activity Diagram for User booking history

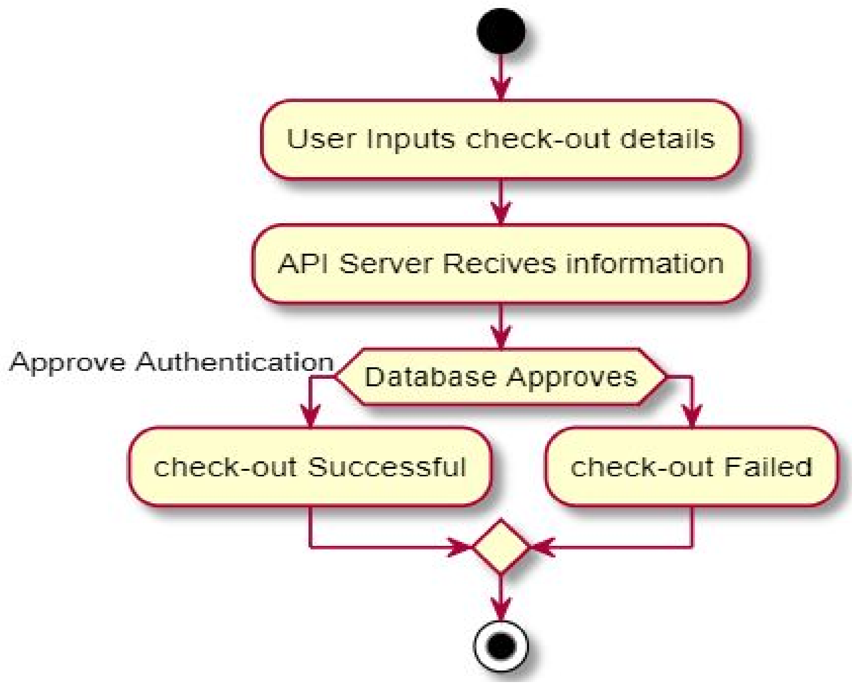
~~~
```plantuml
@startuml
~~~

```
start
:User Inputs booking history details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :booking history Successful;
else
:booking history Failed;
endif
stop

@enduml
```
```


User Inputs booking history details
API Server Recives information
Approve Authentication
Database Approves
booking history Successful
booking history Failed

*4.3.11 Activity Diagram for User check-in*

```
```plantuml
@startuml
start
:User Inputs check-in details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :check-in Successful;
else
:check-in Failed;
endif
stop

@enduml
```
```
```



4.3.12 Activity Diagram for User check-out

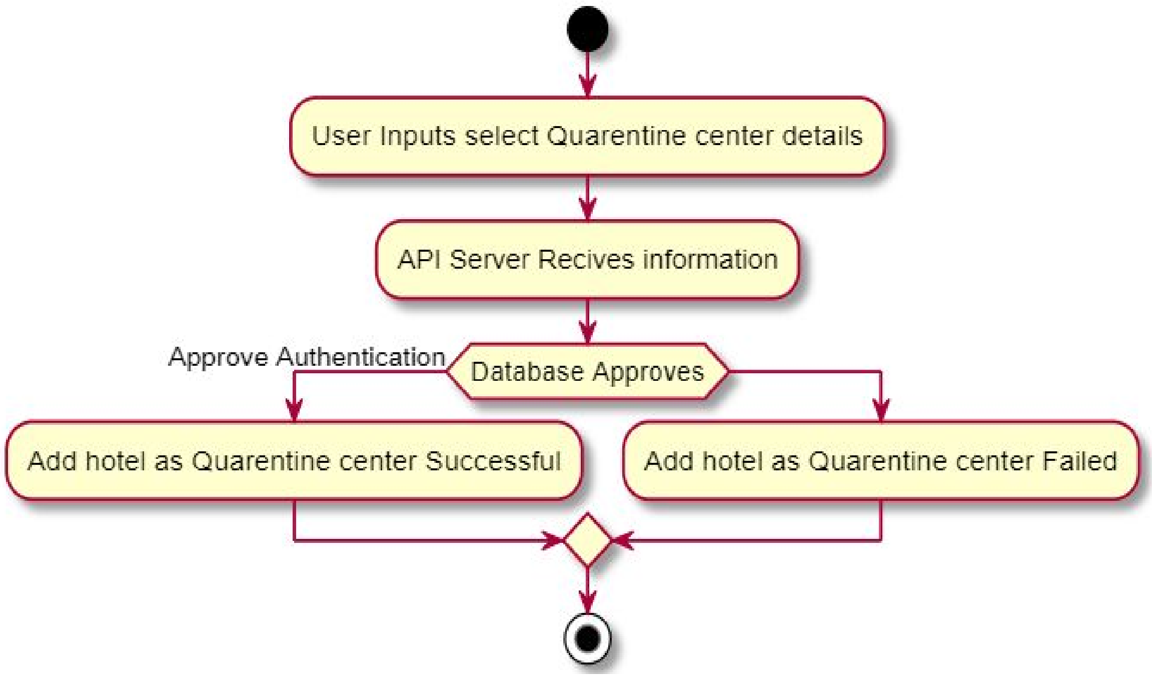
~~~
```plantuml
@startuml
start
:User Inputs check-out details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :check-out Successful;
else
:check-out Failed;
endif
stop

@enduml
```
~~~



4.3.13 Activity Diagram for User add hotel as Quarantine centre

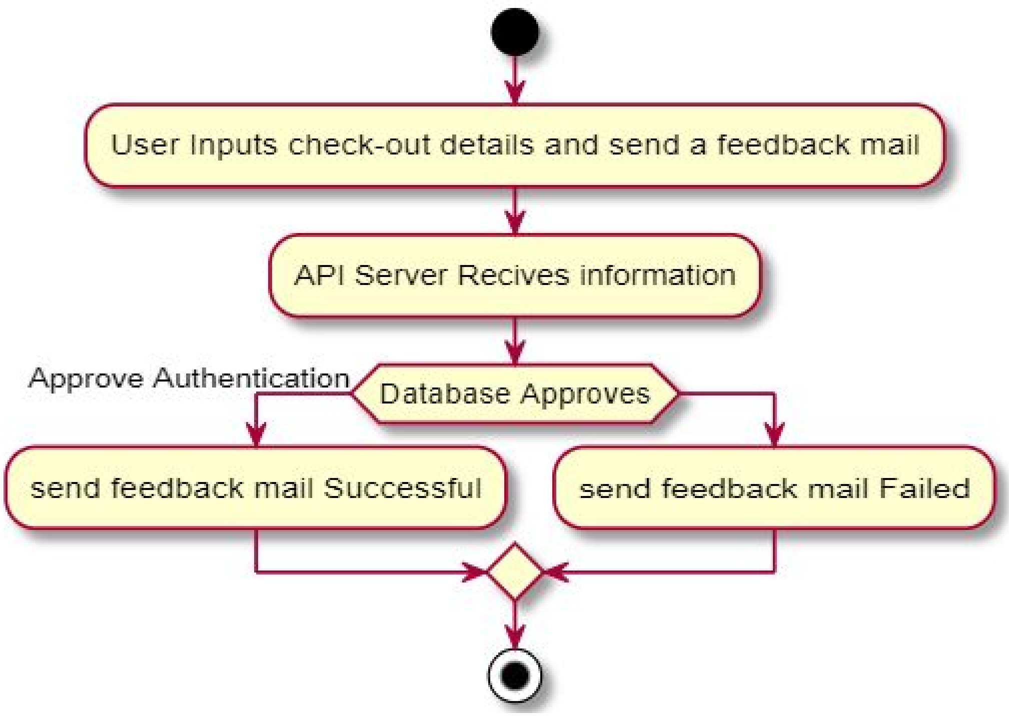
~~~
```plantuml
@startuml
~~~

```
start
:User Inputs select Quarentine center details;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :Add hotel as Quarentine center Successful;
else
:Add hotel as Quarentine center Failed;

endif
stop

@enduml
```


User Inputs select Quarentine center details
API Server Recives information
Approve Authentication
Database Approves
Add hotel as Quarentine center Successful
Add hotel as Quarentine center Failed

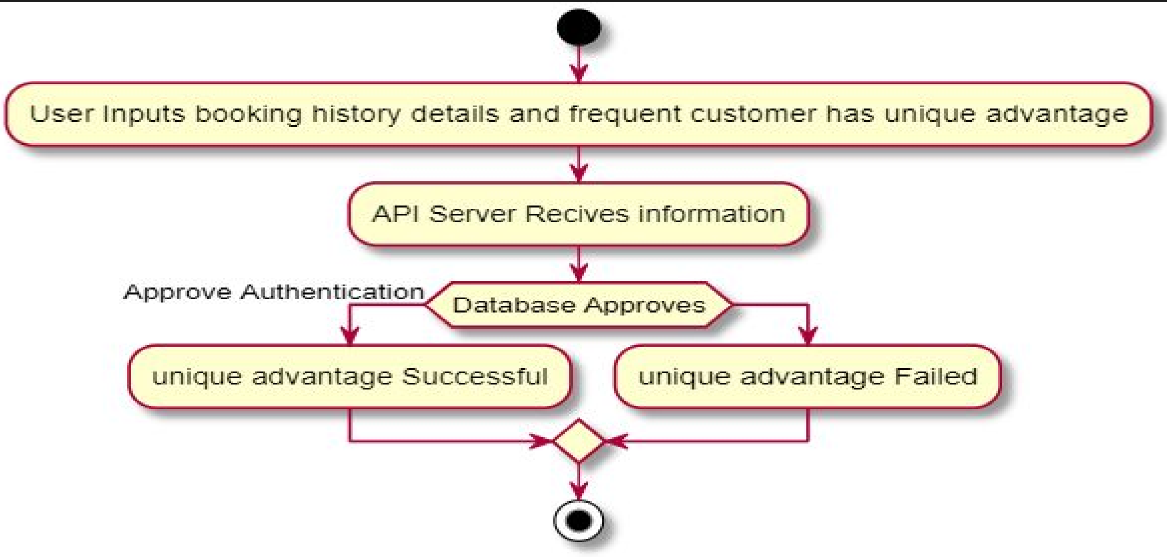
4.3.14 Activity Diagram for User Feedback mail after check-out

~~~
```plantuml
@startuml
start
:User Inputs check-out details and send a feedback mail;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :send feedback mail Successful;
else
:send feedback mail  Failed;
endif
stop

@enduml
```
~~~

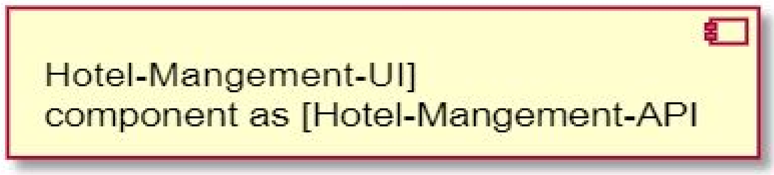

User Inputs check-out details and send a feedback mail
API Server Recives information
Approve Authentication
Database Approves
send feedback mail Successful
send feedback mail Failed

4.3.15 Activity Diagram for User Unique Advantage

~~~
```plantuml
@startuml
start
:User Inputs booking history details and frequent customer has unique advantage;
:API Server Recives information;
if (Database Approves) then (Approve Authentication)
  :unique advantage Successful;
else
:unique advantage  Failed;
endif
stop
@enduml
```
### **3.4 Component Daigram**

```plantuml
~~~


User Inputs booking history details and frequent customer has unique advantage
API Server Recives information
Approve Authentication
Database Approves
unique advantage Successful
unique advantage Failed

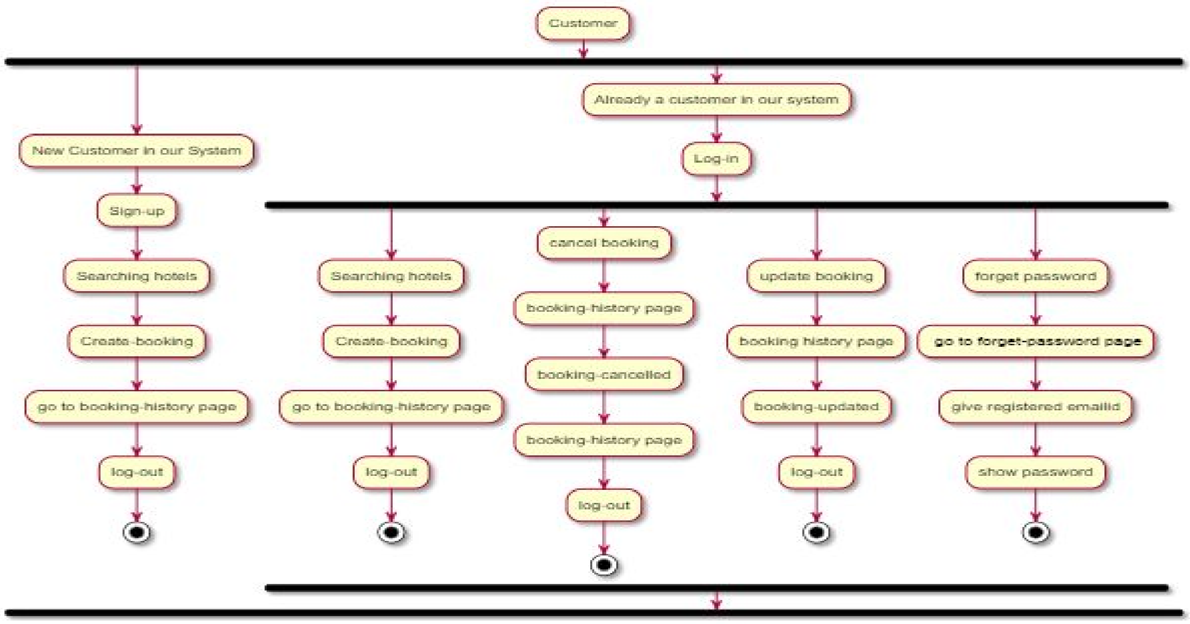
4.4 Component Diagram



4.5 Flowchart

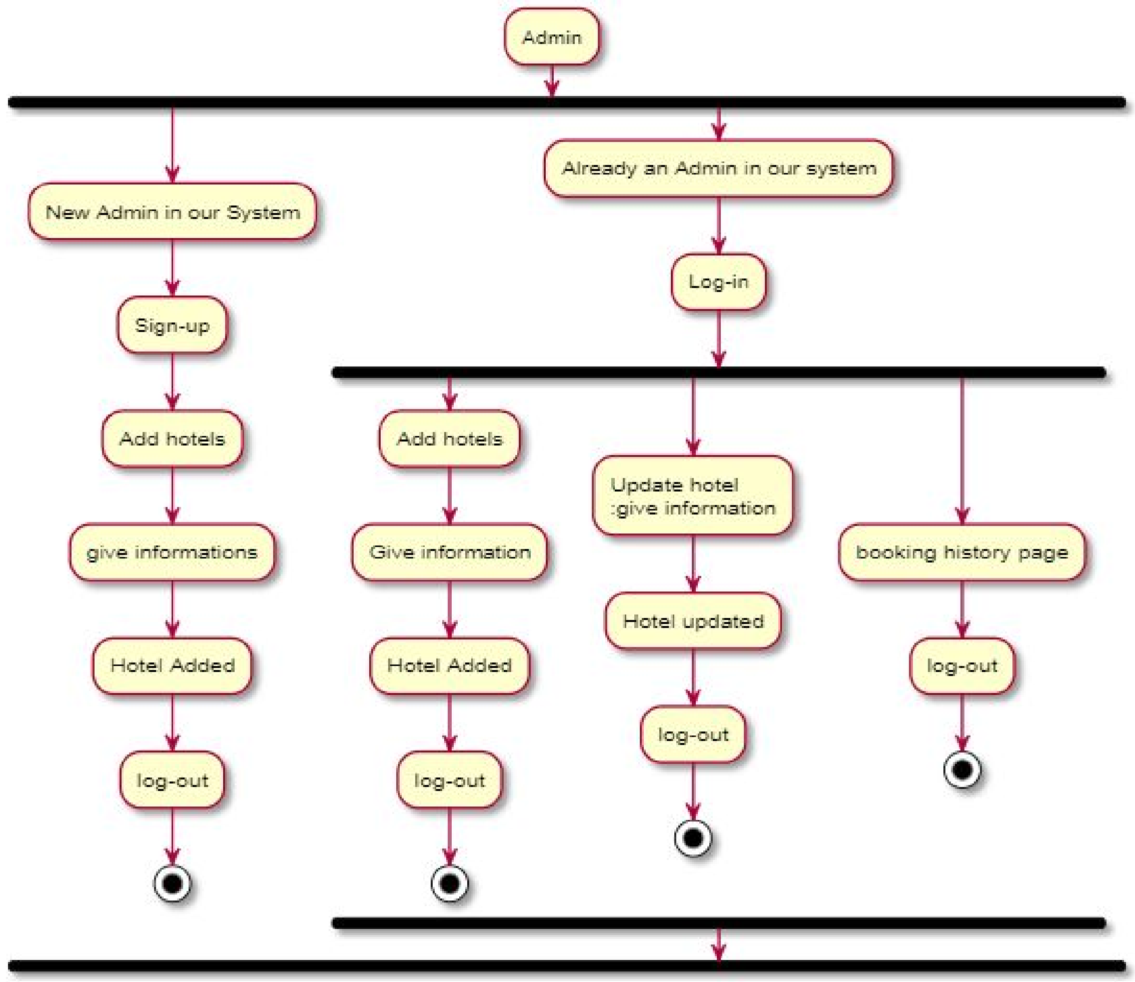
4.5.1 Flowchart for a Customer

~~~
```plantuml
@startuml
:Customer;
   fork
   :New Customer in our System;
   :Sign-up;
   :Searching hotels;
   :Create-booking;
   :go to booking-history page;
   :log-out;
stop
 fork again
   :Already a customer in our system;
   :Log-in;
    fork
   :Searching hotels;
   :Create-booking;
   :go to booking-history page;
   :log-out;
stop
    fork again
     :cancel booking;
     :booking-history page;
     :booking-cancelled;
     :booking-history page;
     :log-out;
stop
    fork again
     :update booking;
    :booking history page;
    :booking-updated;
    :log-out;
stop
    fork again
    :forget password;
    : go to forget-password page;
     :give registered emailid;
     :show password;
stop
@enduml
```
~~~



4.5.2 Flowchart for an Admin in our System

~~~
```plantuml
@startuml
:Admin;
   fork
   :New Admin in our System;
   :Sign-up;
   :Add hotels;
   :give informations;
   :Hotel Added;
   :log-out;
stop
 fork again
   :Already an Admin in our system;
   :Log-in;
    fork
   :Add hotels;
   :Give information;
   :Hotel Added;
   :log-out;
stop
    fork again
     :Update hotel
     :give information;
     :Hotel updated;
     :log-out;
stop
    fork again
    :booking history page;
    :log-out;
stop
@enduml
```
~~~



4.6 User Journey

4.6.1 Customer sign up and login

A User, who is a normal customer for the Hotel Management system that we are creating should be allowed to create a unique session upon Login. If a customer does not have an account on the system, then he should be allowed to sign up with his own unique email id which has used for registration previously. He should add the details requested along with a valid 10-digit mobile number and a password that is Alphanumeric and contains at least 8 letters, a special character and 3 digits.

4.6.2 Admin Signup and login

A User, who is a normal Admin for the Hotel Management system that we are creating should be allowed to create a unique session upon Login. If a customer does not have an account on the system, then he should be allowed to sign up with his own unique email id which has used for registration previously. He should add the details requested along with a valid 10-digit mobile number and a password that is Alphanumeric and contains at least 8 letters, a special character and 3 digits.

4.6.3 Customer logout

A User, who is a customer for the Hotel Management system want to logout from our website simply click on logout button, a message is shown in our website he is logged-out successfully. Then, when he can book any hotel through our website in future, he must be signup in our website as above-mentioned process, it is mandatory for all customer.

4.6.4 Admin logout

A User, who is an Admin for the Hotel Management system want to logout from our website simply click on logout button, a message is shown in our website he is logged-out successfully. Then, when he can book any hotel through our website in future, he must be signup in our website as above-mentioned sign process, it is mandatory for all customer.

4.6.5 Customer create booking, update booking, cancel booking

A User, who is a customer for the Hotel Management system want to book a hotel in any place, at first, he can choice any hotel, if he founds a Quarantine centre then it is available in our system also. He choices a hotel and sees the hotel rent per day then goes to the create booking page and fills the all necessary information's, he adds his name, registered emailid,contact number etc and books a hotel sucessfully.After,he wants to update his booking (change hotel, change booking date etc)simply goes to the update booking page and then fills all information's or change some information and updates booking successfully Customer somehow wants to cancel his booking goes to the cancel booking page and simply clicks on cancel booking link and a message is shown in our website and then checks the booking history page to see there must be no booking is available there.

4.6.6 Admin Add hotel, update hotel, manage hotel

A User, who is an Admin for the Hotel Management system want to add a hotel in any place, at first he wants to add any hotel, if he adds this hotel as a Quarantine centre then selects the option Quarantine centre .He adds a hotel and also add the hotel rent per day then goes to the add hotel page and fills the all necessary information's, he adds his name, registered emailid,contact number etc and adds a hotel sucessfully.After,he wants to update his hotel or add some new hotel(change hotel, change anything etc)simply goes to the update hotel page and then fills all information's or change some information and updates hotel successfully Admin always sees the customer's booking history to go to the booking history page to manage his hotel's room free in customer's check-in time and also gave reminder to customer in his check-out time.

4.6.7 Customer check in and check out

Check-in is used to admit a customer in our Hotel after entering his all personal details like Name, Address, Phone, Sex and then he/she is assigned a room. Check-out is used to check out the customer details from database. When the user inputs his room number, the same room number will be checked in the database, if the room number is matched in the database, then the customer will be check-out from the database and transferred the record of the checkout to another table of database so that the Hotel Management has the record of customers who have check-out to fulfil his legal liabilities.

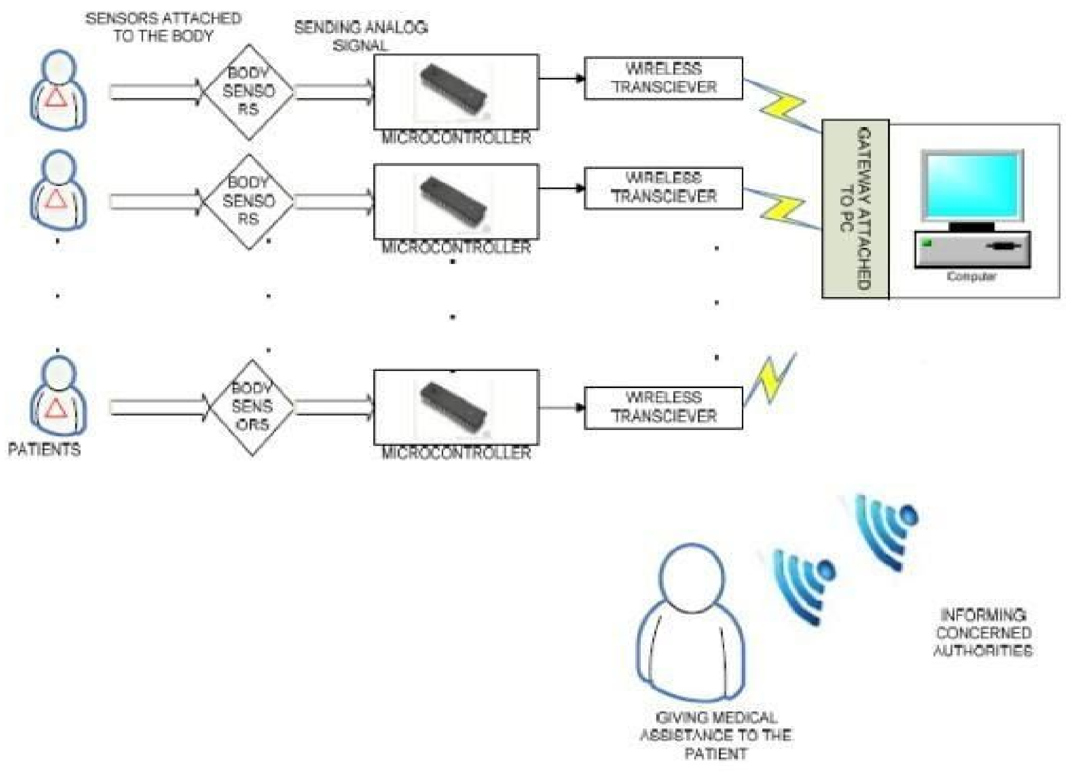
4.6.8 Listing of available hotels

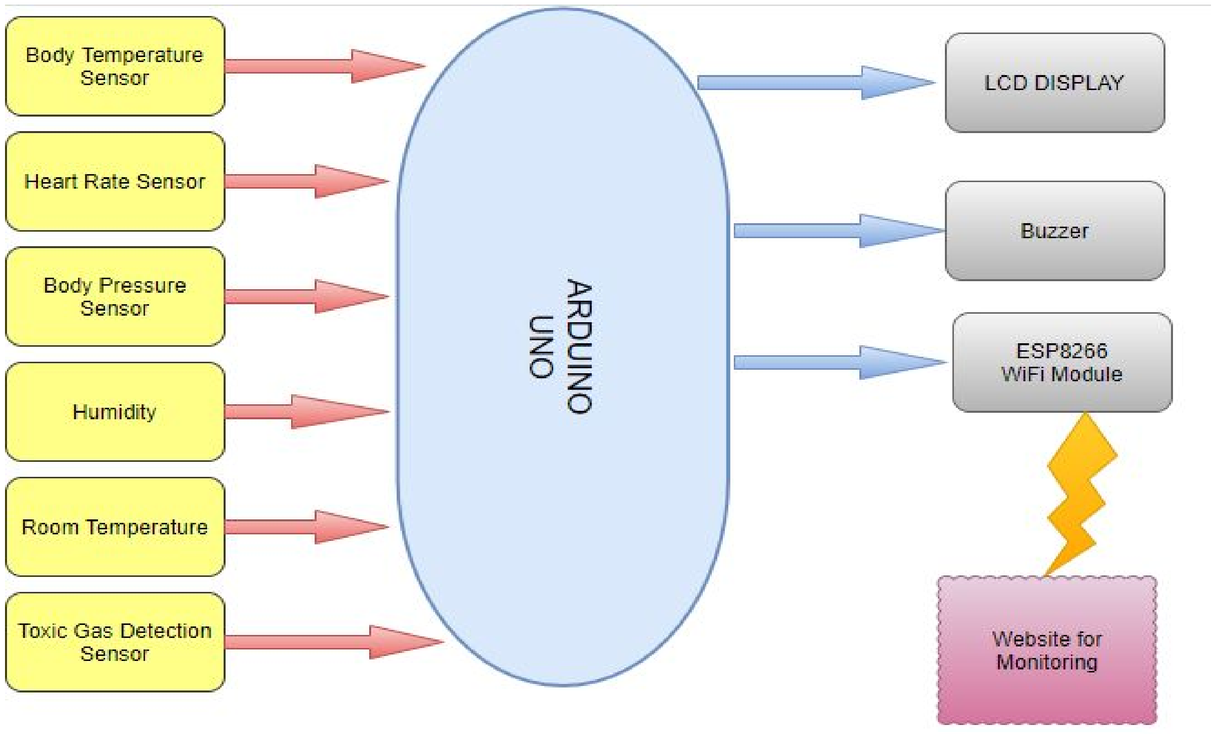
A User, who is a customer for the Hotel Management system want to see list of available hotels in different cities, different places simply go to our hotel front page and choose any place as his wish, there many places hotels are available and see all hotels images,name,rent per day ,and also see the hotel is either Quarantine centre or not.

5. Work Flow (STAGE -2)

5.1 Work Done

Here we have used active network technology to network various sensors to a single system. Patients' various critical parameters are continuously monitored via single system and reported to the Doctors or Nurses in attendance for timely response in case of critical situations. The sensors are attached to the body of the patients without causing any discomfort to them. In this system we monitor the important physical parameters like body temperature, heart beat rate and blood pressure using the sensors which are readily available. Thus, the analog values that are sensed by the different sensors are then given to a microcontroller attached to it. The microcontroller processes these analog signal values of health parameters separately and converts it to digital values using ADC converter. Now, the digitalized values from more than one microcontroller are sent to the Central System. Each of the sensors attached microcontroller with a transceiver will act as a module which has its own unique ID. Each module transmits the data wirelessly to the gateway attached to the PC of the Central System. The gateway is attached to the PC i.e. Central System which is situated in the medical center, is capable for selecting different patient IDs and allowing the gateway to receive different physical parameter values the patient specified by the ID .At any time, any of the doctors or nurses can log on the Central System and check the history of the observed critical parameters of any of the patient attached to the network.





5.2 Sensors and Modules

1. Arduino Microcontroller
2. ESP866 Wi-Fi Module
3. Heart Rate Sensor
4. BS18B20 Body Temperature sensor
5. DHT 11 Humidity Sensor
6. Toxic Gas sensor
7. Air Quality sensor

8.Beard Board

9 16*2 Display

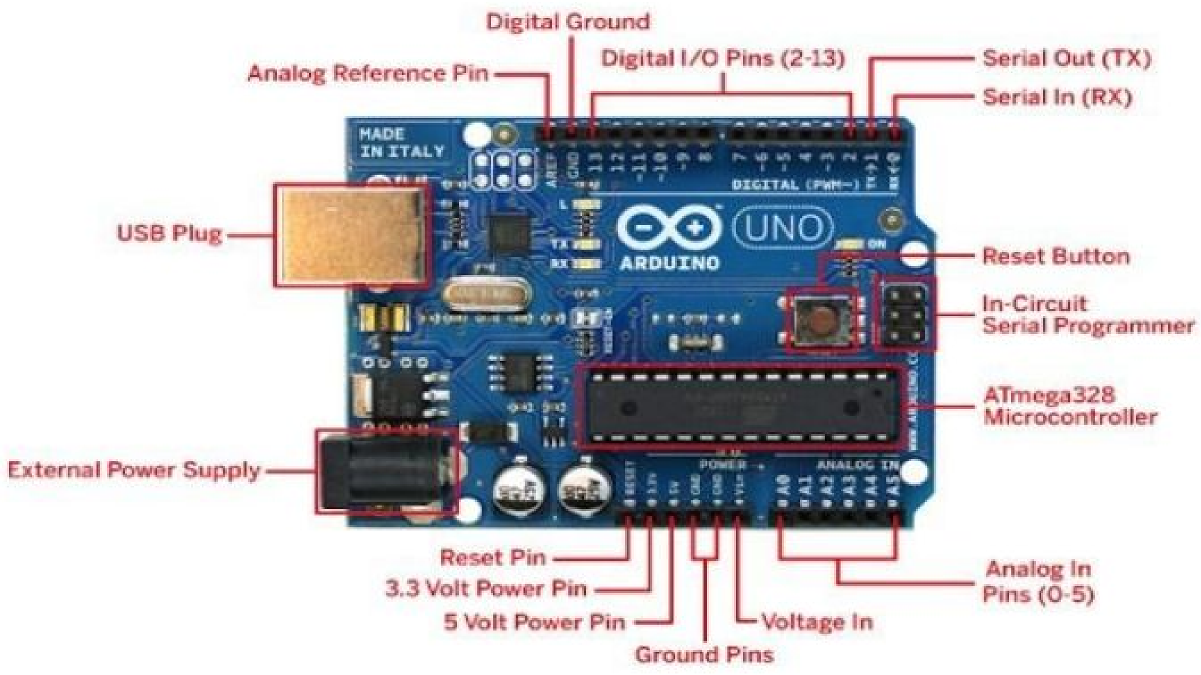
10.Buzzer

11.Male to Female Wire

12.Resistor

5.2.1 Arduino Microcontroller

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.



Starting clockwise from the top center:

1. Analog Reference pin

2. Digital Ground

3. Digital Pins 2-13

4. Digital Pins 0-1/Serial In/Out - TX/RX (dark green) - These pins cannot be used for digital i/o (digital Read and digital Write) if you are also using serial communication (e.g. Serial. Begin).

5. Reset Button

6. In-circuit Serial Programmer

7. Analog in Pins 0-5

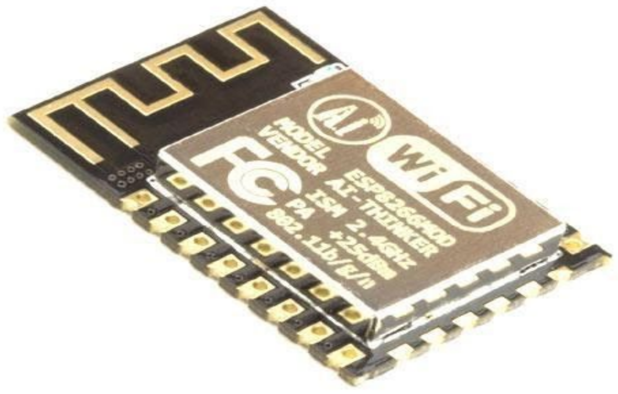
8. Power and Ground Pins

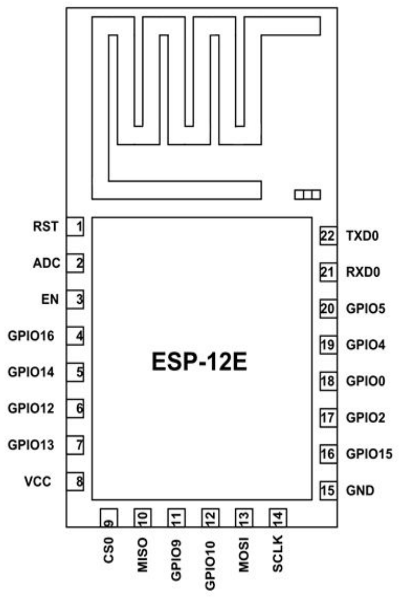
9. External Power Supply In (9-12VDC)

10. Toggles External Power and USB Power (place jumper on two pins closest to desired supply)

11. USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board)

5.2.2 ESP8266 Wi-Fi Module





ESP-12E is a miniature **Wi-Fi module** present in the market and is used for establishing a wireless network connection for microcontroller or processor. The core of ESP-12E is **ESP8266EX**, which is a high integration wireless SoC (System on Chip). It features ability to embed Wi-Fi capabilities to systems or to function as a standalone application. It is a low-cost solution for developing IoT applications.

Pin Configuration -The ESP-12E module has twenty-two pins and we will describe function of each pin below.

Pin	**Name**	**Description**
1	RST	Reset Pin of the module
2	ADC	Analog Input Pin for 10-bit ADC (0V to1V)
3	EN	Module Enable Pin (Active HIGH)
4	GPIO16	General Purpose Input Output Pin 16
5	GPIO14	General Purpose Input Output Pin 14

6	GPIO12	General Purpose Input Output Pin 12
7	GPIO13	General Purpose Input Output Pin 13
8	VDD	+3.3V Power Input
9	CS0	Chip selection Pin of SPI interface
10	MISO	MISO Pin of SPI interface
11	GPIO9	General Purpose Input Output Pin 9
12	GPIO10	General Purpose Input Output Pin 10
13	MOSI	MOSI Pin of SPI interface
14	SCLK	Clock Pin of SPI interface
15	GND	Ground Pin
16	GPIO15	General Purpose Input Output Pin 15
17	GPIO2	General Purpose Input Output Pin 2
18	GPIO0	General Purpose Input Output Pin 0
19	GPIO4	General Purpose Input Output Pin 4
20	GPIO5	General Purpose Input Output Pin 5
21	RXD0	UART0 RXD Pin
22	TXD0	UART0 TXD Pin

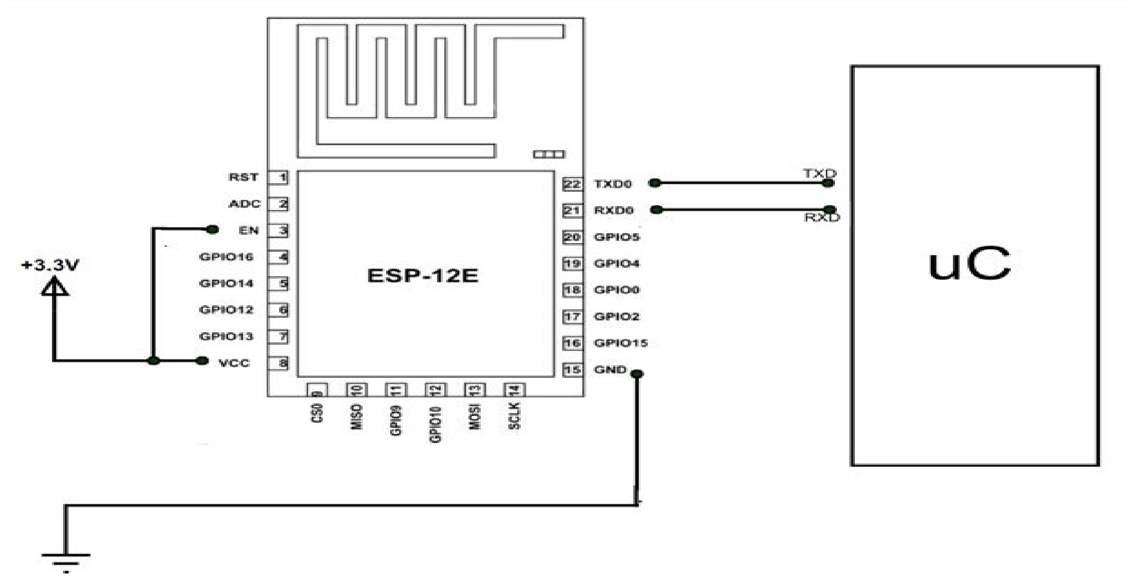
Features and Electrical Characteristics

- Wireless Standard: IEEE 802.11 b/g/n protocol
- Power Transmission:

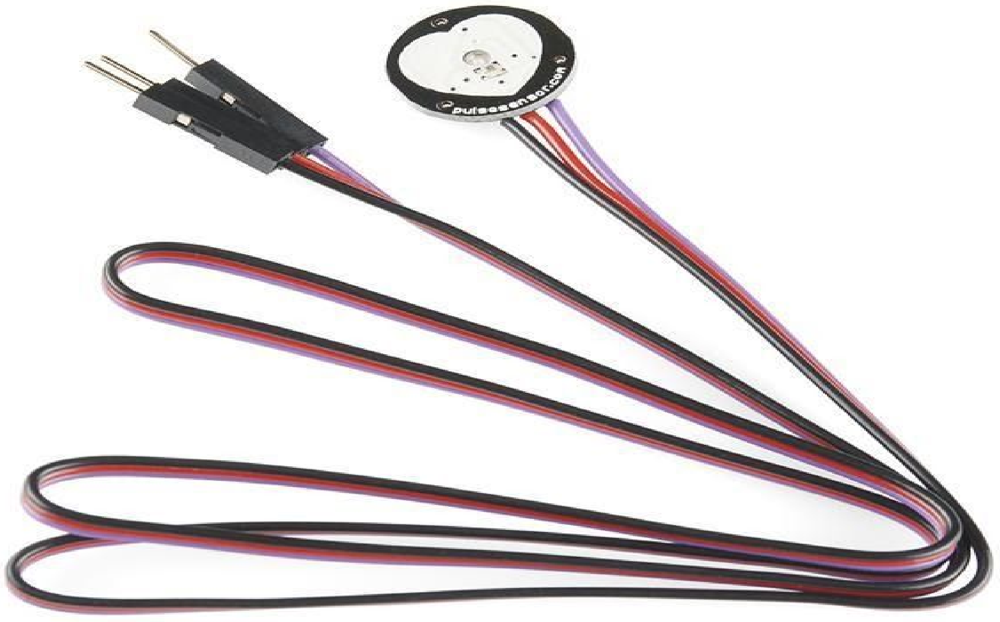
802.11b	+16 ± 2 dBm
802.11g	+14 ± 2 dBm
802.11n	+13 ± 2 dBm

- Frequency Range: 2.412 - 2.484 GHz

- Serial Transmission: 110 - 921600 bps, TCP Client 5
- SDIO 2.0, SPI and UART Interface available
- PWM available
- One ADC channel available
- Programmable GPIO available
- Wireless Network Type: STA / AP / STA + AP
- Security Type: WEP / WPA-PSK / WPA2-PSK
- Encryption Type: WEP64 / WEP128 / TKIP / AES
- Network Protocol: IPv4, TCP / UDP / FTP / HTTP
- Operating Voltage: 3.3V
- Maximum current allowed to draw per pin: 15mA
- Power down leakage current of < 10uA
- Integrated low power 32-bit MCU
- Onboard PCB Antenna
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW
- Operating Temperature: -40ºC to +125 ºC



5.2.3 Heart Rate Sensor



The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects. It essentially combines a simple optical heart rate sensor with amplification and noise cancellation circuitry making it fast and easy to get reliable pulse readings. Also, it sips power with just 4mA current draw at 5V so it's great for mobile applications.

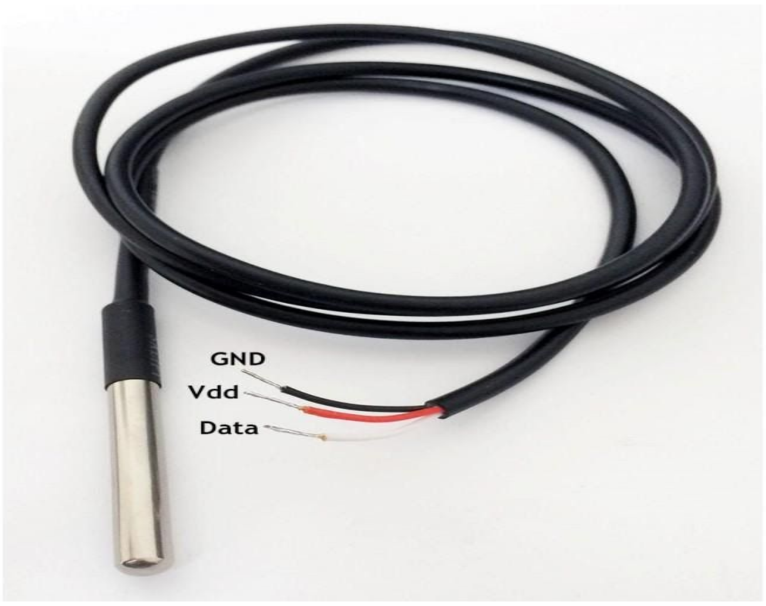
Live Heartbeats and BPM in your Project. Works with Arduino, micro: bit, little Bits, and most maker platforms.

Simply clip the Pulse Sensor to your earlobe or fingertip and plug it into your 3 *or *5 Volt Arduino and you're ready to read heart rate!
It has only 3 pins, VCC, GND and Signal Pin.

Features:

- Biometric Pulse Rate or Heart Rate detecting sensor
- Plug and Play type sensor
- Operating Voltage: +5V or +3.3V
- Current Consumption: 4mA
- Inbuilt Amplification and Noise cancellation circuit.
- Diameter: 0.625”
- Thickness: 0.125” Thick

5.2.4 BS18B20 Body Temperature Sensor



This is a 1 Meter Long Waterproof, sealed and pre-wired digital temperature sensor probe based on DS18B20 sensor. It is very handy for when you need to measure something far away, or in wet conditions. Because they are digital, you don't get any signal degradation even over long distance.

These 1-wire digital temperature sensors are fairly precise (±0.5°C over much of the range) and can give up to 12 bits of precision from the onboard digital-to-analog converter. They work great with any microcontroller using a single digital pin, and you can even connect multiple ones to the same pin, each one has a unique 64-bit ID burned in at the factory to differentiate them. Usable with 3.0-5.0V systems.

The only downside is they use the Dallas 1-Wire protocol, which is somewhat complex, and requires a bunch of code to parse out the communication. When using with microcontroller put a 4.7k resistor to sensing pin, which is required as a pullup from the DATA to VCC line.

Cable specs: -

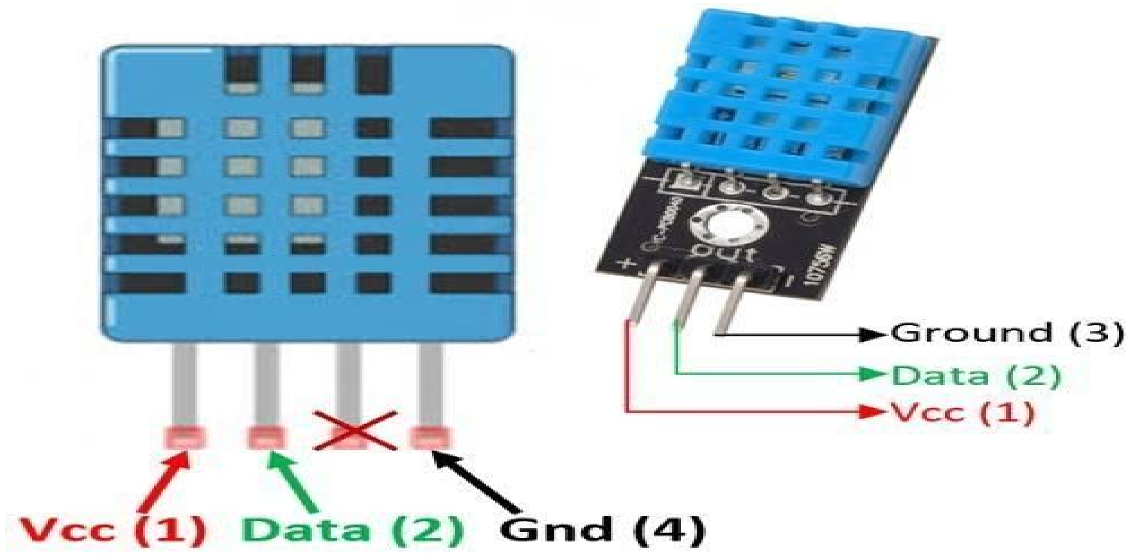
- Stainless steel tube 6mm diameter by 30mm long
- Cable is 36" long / 91cm, 4mm diameter (1 Meter Long)
- Contains DS18B20 temperature sensor
- Three wires - Red connects to 3-5V, Black connects to ground and White is data.

DS18B20 Sensor Technical specs: -

- Usable temperature range: -55 to 125°C (-67°F to +257°F)

- 9 to 12-bit selectable resolution
- Uses 1-Wire interface- requires only one digital pin for communication
- Unique 64-bit ID burned into chip
- Multiple sensors can share one pin
- ±0.5°C Accuracy from -10°C to +85°C
- Temperature-limit alarm system
- Query time is less than 750ms
- Usable with 3.0V to 5.5V power/data

5.2.5 DHT 11 Humidity Sensor



DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. DHT11 can be interface with any microcontroller like Arduino, Raspberry Pi, etc. and get instantaneous results. DHT11 is a low-cost humidity and temperature sensor which provides high reliability and long-term stability.

It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and outputs a digital signal on the data pin (no analog input pins needed). It's very simple to use, and libraries and sample codes are available for Arduino and Raspberry Pi.

This module makes is easy to connect the DHT11 sensor to an Arduino or microcontroller as includes the pull up resistor required to use the sensor. Only three connections are required to be made to use the sensor - Vcc, Gnd and Output.

It has high reliability and excellent long-term stability, thanks to the exclusive digital signal acquisition technique and temperature & humidity sensing technology.

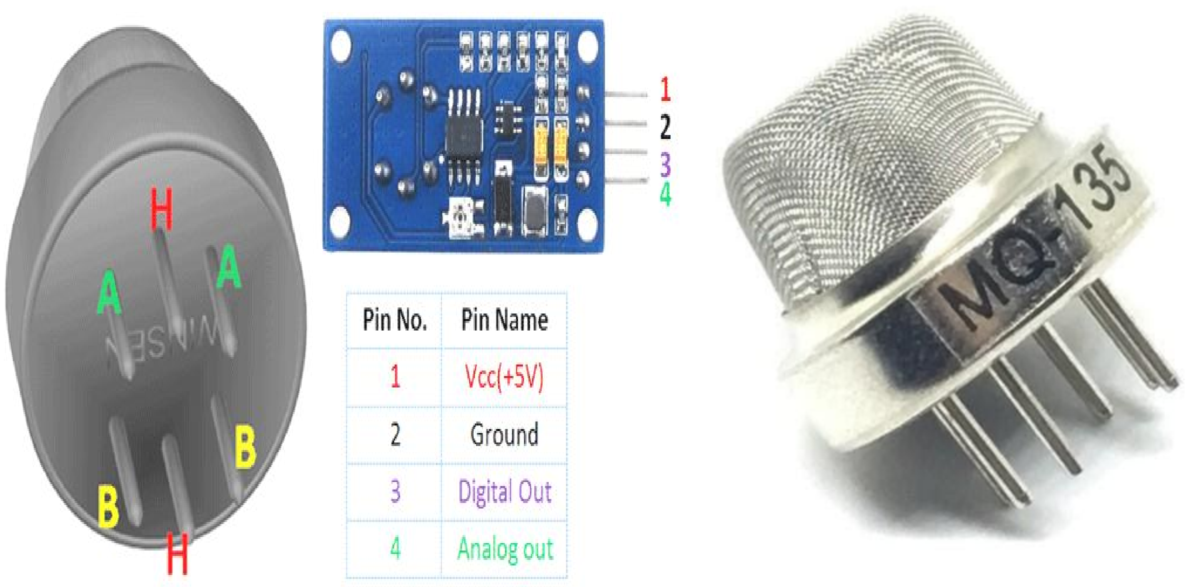
Specifications: -

- Power Supply： 3.3~5.5V DC
- Output： 4 pin single row
- Measurement Range： Humidity 20-90%RH， Temperature 0~50℃
- Accuracy： Humidity +-5%RH， Temperature +-2℃
- Resolution： Humidity 1%RH， Temperature 1℃
- Interchangeability： Fully Interchangeable
- Long-Term Stability： <±1%RH/year

Pin Description: -

- Pin 1: Power +Ve (3.3VDC to 5.5VDC Max wrt. GND)
- Pin 2: Serial Data Output
- Pin 3: Power Ground or Power –Ve

5.2.6 Toxic Gas/Air Quality Sensor



Pin No.	Pin Name
1	Vcc(+5V)
2	Ground
3	Digital Out
4	Analog out

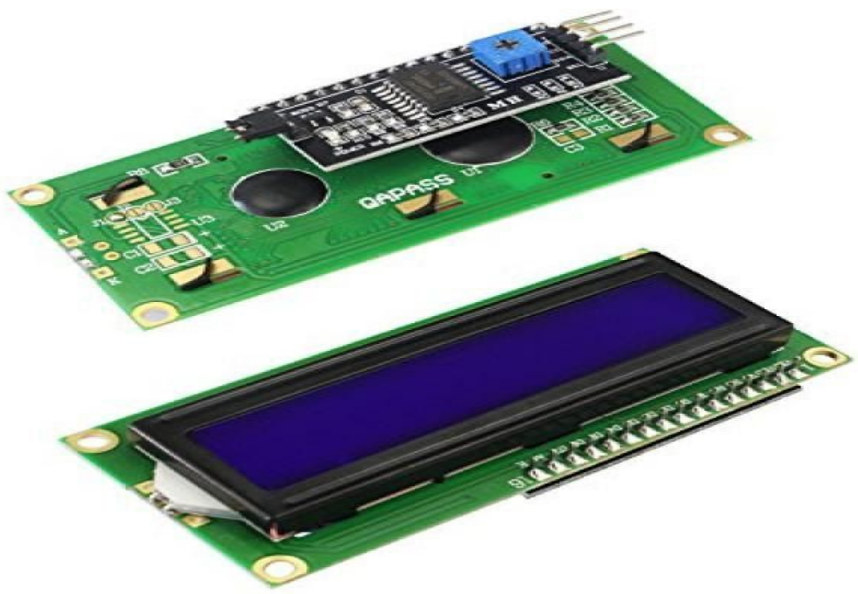
MQ135 Gas Sensor module for Air Quality having Digital as well as Analog output. Sensitive material of MQ135 gas sensor is SnO2, which with lower conductivity in clean air. When the target combustible gas exists, the sensors conductivity is higher along with the gas concentration rising.

MQ135 gas sensor has high sensitivity to Ammonia, Sulphide and Benzene steam, also sensitive to smoke and other harmful gases. It is with low cost and suitable for different application.
Used for family, surrounding environment noxious gas detection device, apply to ammonia, aromatics, sulphur, benzene vapor, and other harmful gases/smoke, gas detection, tested concentration range: 10 to 1000 ppm.

Specifications of MQ135 Gas Sensor Module: -

- Working voltage: DC 5V
- Working Current: 150mA
- DOUT: TTL output
- AOUT: Analog output
- Preheat time: Over 20s
- Dimension: 32mm x 22m x 27mm (HIGH 27mm)

5.2.7 16*2 LCD Display



This is LCD 1602 Parallel LCD Display that provides a simple and cost-effective solution for adding a 16×2 White on Liquid Crystal Display into your project. The display is 16 character by 2-line display has a very clear and high contrast white text upon a blue background/backlight. This is great blue backlight LCD display. It is fantastic for Arduino based project. This LCD1602 LCD Display is very easy to interface with Arduino or Other Microcontrollers.

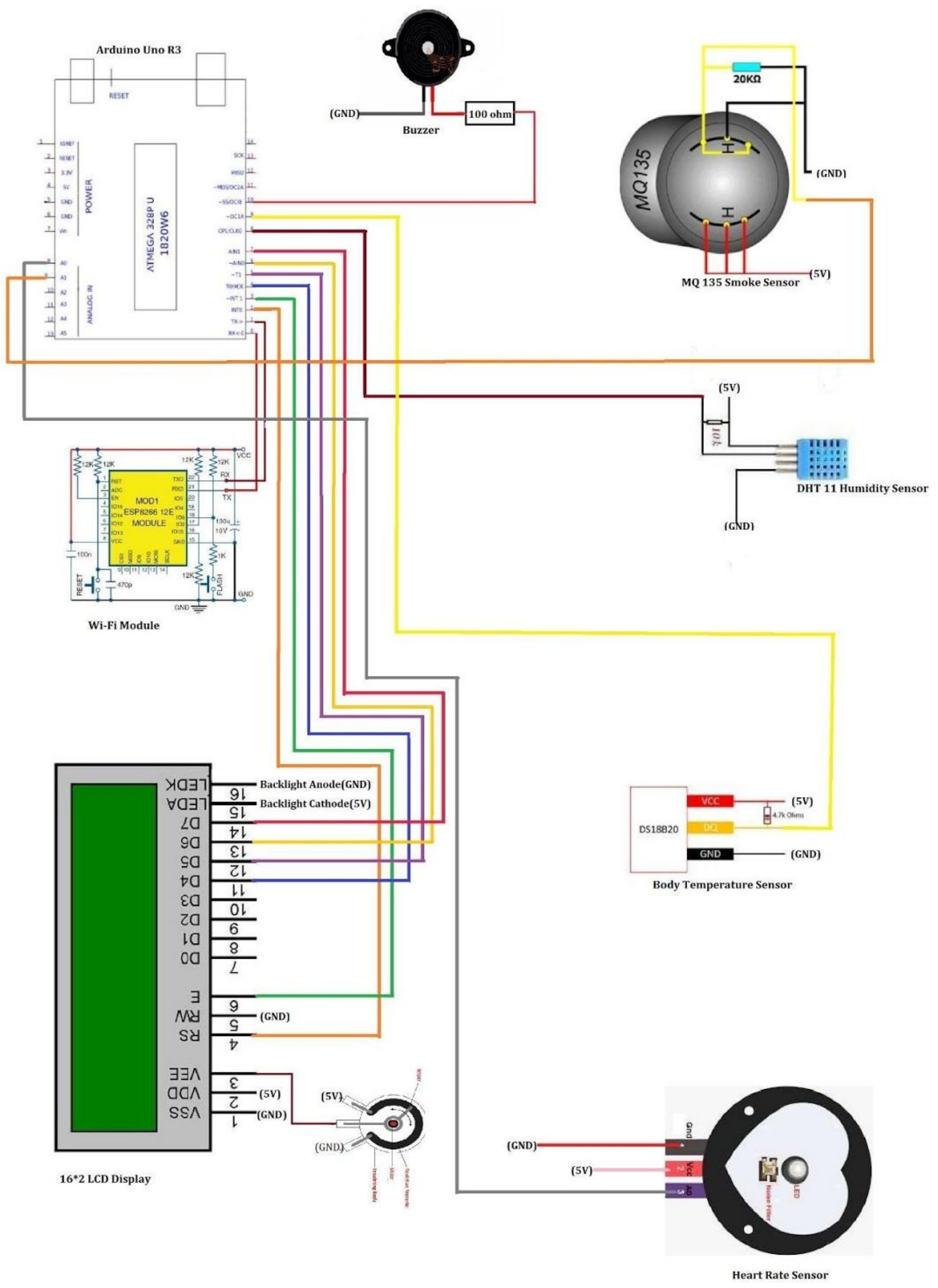
This display overcomes the drawback of LCD 1602 Parallel LCD Display in which you'll waste about 8 Pins on your Arduino for the display to get working. Luckily in this product, an I2C adapter is directly soldered right onto the pins of the display. So, all you need to connect are the I2C pins, which shows a good library and little of coding.

The I2C is a type of serial bus developed by Philips, which uses two bidirectional lines, called SDA (Serial Data Line) and SCL (Serial Clock Line). Both must be connected via pulled-up resistors. The usage voltages are standard as 5V and 3.3V.If you already have the I2C adapter soldered onto the board like in this product, the wiring is quite easy. You should usually have only four pins to hook up. VCC and GND of course. The LCD display works with 5 Volts. So, we go for the 5V Pin.The values shown on the display can be either a simple text or numerical values read by the sensors, such as temperature or pressure, or even the number of cycles that the Arduino is performing.

Specifications & Features: -

- Arduino IIC/I2C interface was developed to reduce the IO port usage on Arduino board
- I2C adapter allows flexibility in connections
- I2C Reduces the overall wirings.
- 16 characters wide, 2 rows
- White text on the Blue background
- Single LED backlight included can be dimmed easily with a resistor or PWM.
- Interface: I2C
- Interface Address: 0x27
- Character Colour: White
- Backlight: Blue
- Supply voltage: 5V

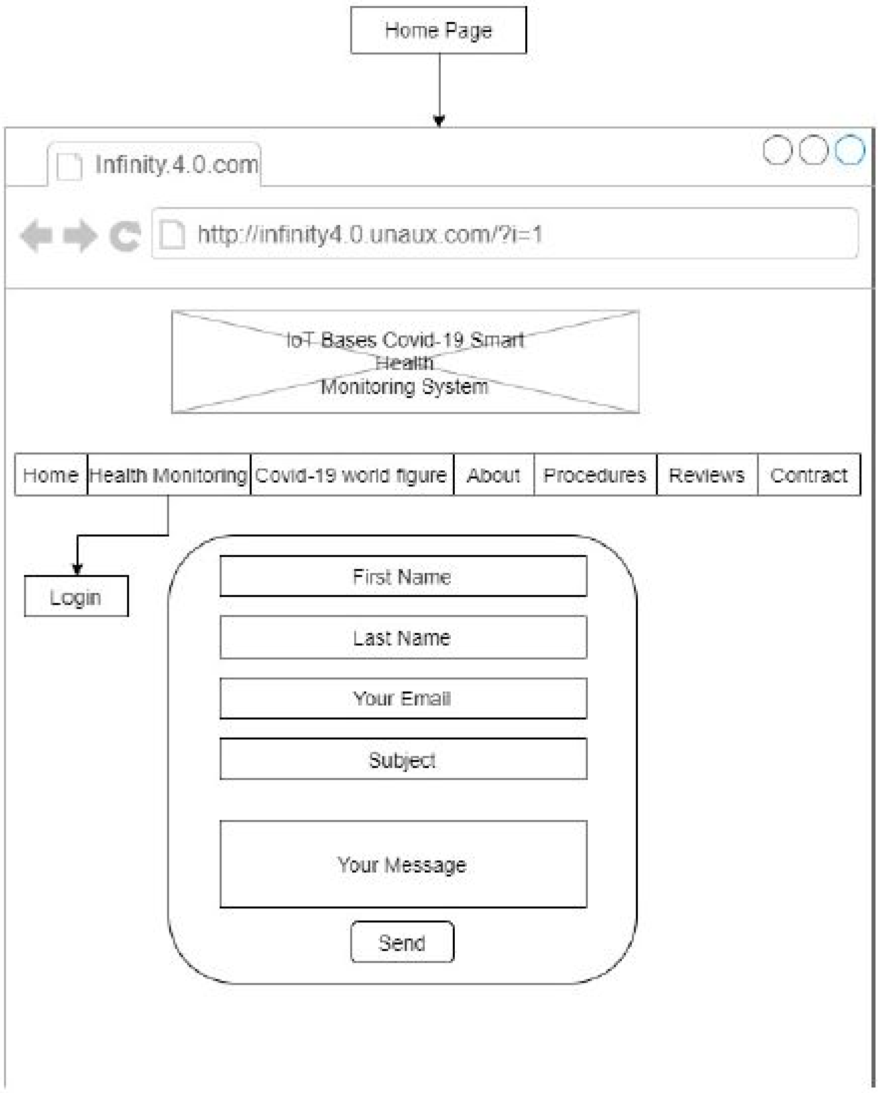
5.3 Circuit Diagram

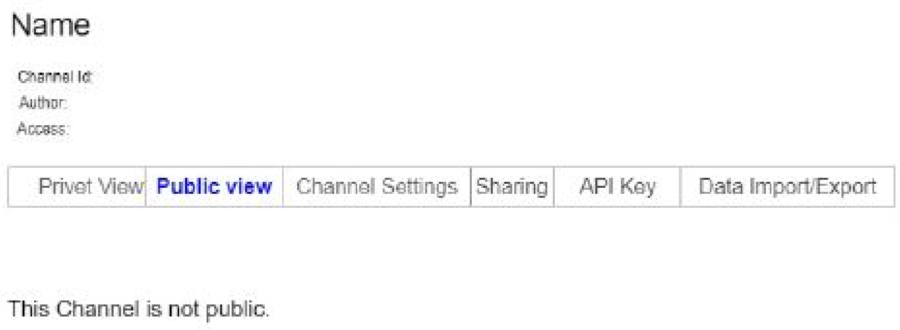


Arduino Uno R3
RESET
ATMEGA 328P U
1820W6
POWER
ANALOG IN
(GND)
Buzzer
100 ohm
20KΩ
MQ135
(GND)
(5V)
MQ 135 Smoke Sensor
(5V)
10k
DHT 11 Humidity Sensor
(GND)
VCC
RX
TX
MOD1
ESP8266 12E
MODULE
100u
10V
100n
470p
RESET
FLASH
GND
Wi-Fi Module
LEDK
LEDA
D7
D6
D5
D4
D3
D2
D1
D0
E
RW
RS
VEE
VDD
VSS
Backlight Anode(GND)
Backlight Cathode(5V)
(GND)
(5V)
(GND)
(5V)
(GND)
16*2 LCD Display
VCC
DQ
GND
DS18B20
(5V)
4.7k Ohms
(GND)
Body Temperature Sensor
(GND)
(5V)
Gnd
Vcc
A0
LED
Noise Filter
Heart Rate Sensor

5.4 Health Parameters Monitoring Via ThingSpeak & Wireframe

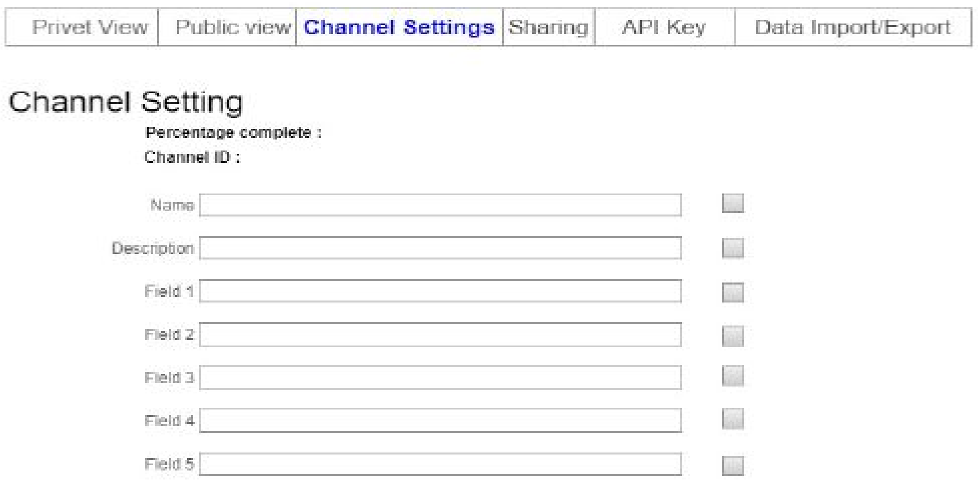
ThingSpeak **is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP and MQTT protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates.**





This Channel is not public.

(The health parameters view can be set to private/public)



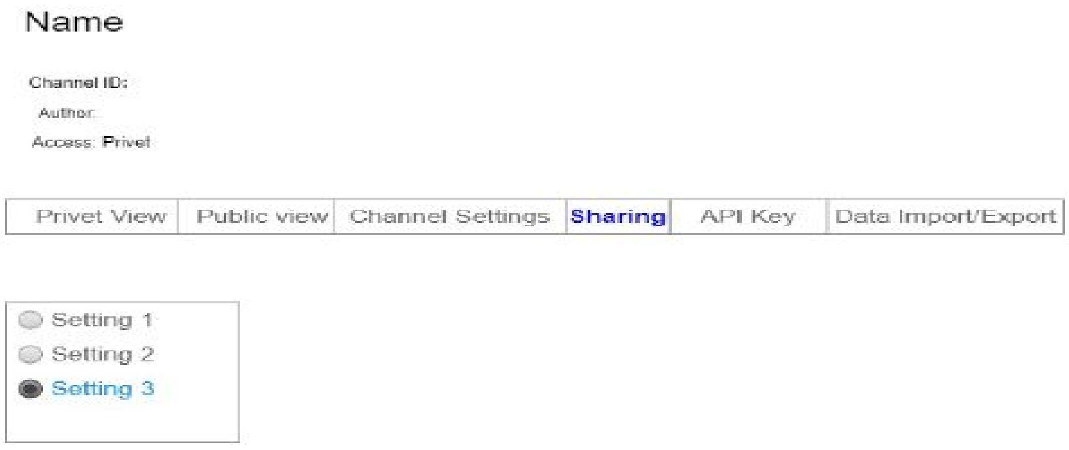
Channel Setting

Percentage complete :

Channel ID :

Name	
Description	
Field 1	
Field 2	
Field 3	
Field 4	
Field 5	

(The no of fields denotes the reading of the sensors to be collected)



Name

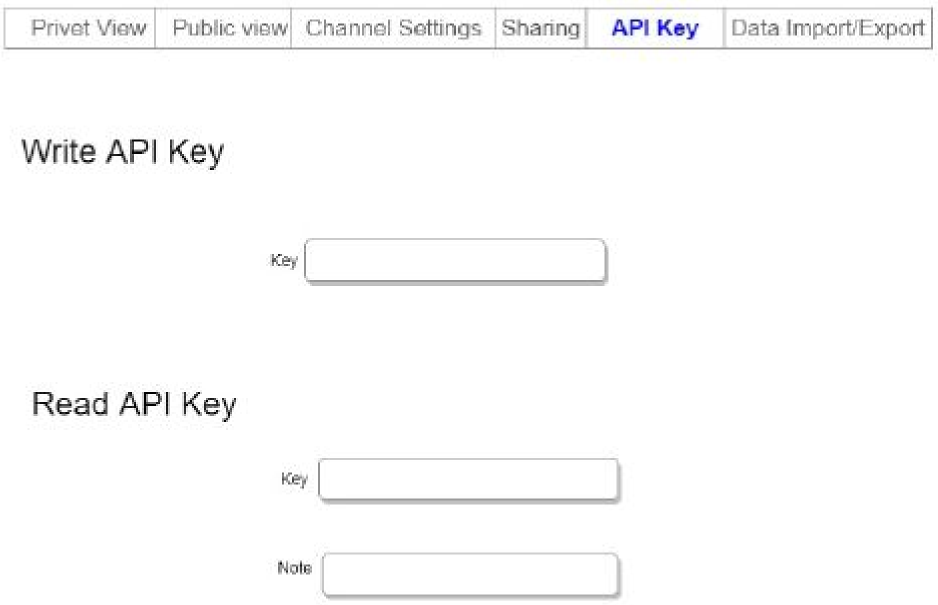
Channel ID:

Author:

Access: Privet

○ Setting 1
○ Setting 2
● Setting 3

(The data can be also shared with particular person)



Write API Key

Key

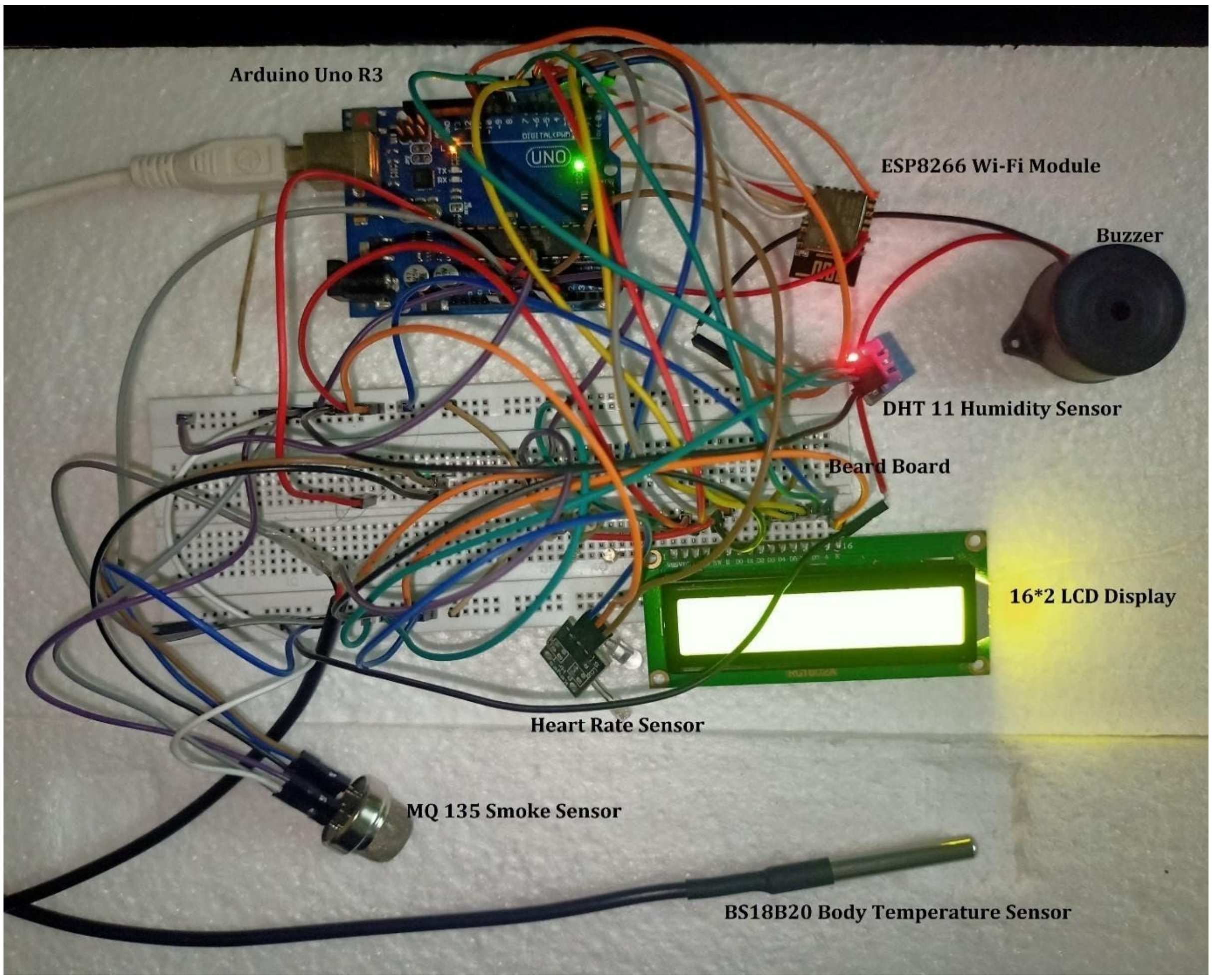
Read API Key

Key

Note

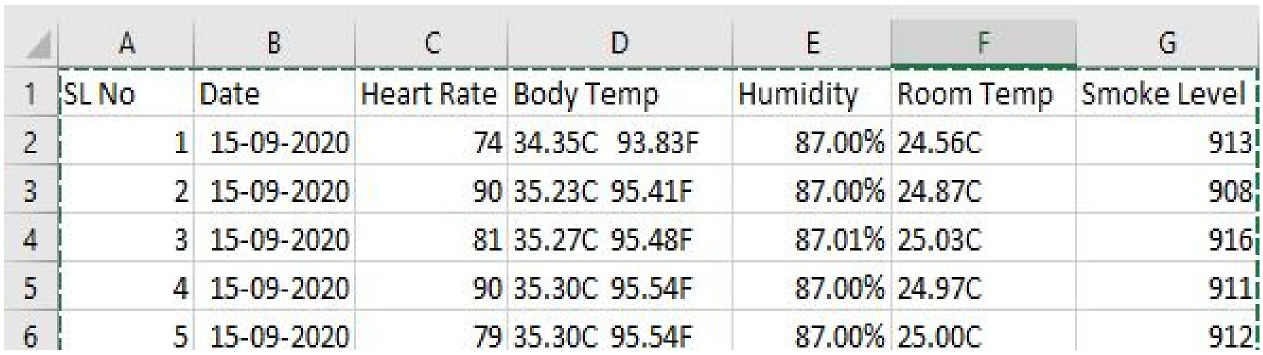
(API Key
Used
To transfer the data)

5.5 System Setup



In the above image, it is shown complete device setup which includes Arduino micro controller board with power supply attached to it. Micro controller is connected with all the sensors which includes Air quality sensor, Toxic gas sensor, Humidity sensor, Temperature sensor, Heartbeat sensor. Micro controller also connected with alarm which will be used in case any of sensor data conditions are not met like temperature spikes, toxic gases etc. Wi-Fi module is connected which are used to send sensor data into server. LCD is connected to micro controller which displays series of information as soon as device is turned on and finally, once device is connected to network, it displays all the patient information on it along with any irregularities of patient vitals.

5.6 Results



	A	B	C	D	E	F	G
1	SL No	Date	Heart Rate	Body Temp	Humidity	Room Temp	Smoke Level
2	1	15-09-2020	74	34.35C 93.83F	87.00%	24.56C	913
3	2	15-09-2020	90	35.23C 95.41F	87.00%	24.87C	908
4	3	15-09-2020	81	35.27C 95.48F	87.01%	25.03C	916
5	4	15-09-2020	90	35.30C 95.54F	87.00%	24.97C	911
6	5	15-09-2020	79	35.30C 95.54F	87.00%	25.00C	912

5. Work Flow (STAGE -3)

Initially we started our project by finding out the symptoms of COVID-19 patients that are most common.

The symptoms are listed below

Most Common Symptoms:

Fever / chill

Breathlessness

Dry Cough

Tiredness

Less Common Symptoms:

Fatigue

Muscle pain

Mucus formation

Clogged nasal cavity

Sore throat

Diarrhoea

Loss of taste (anosmia).

Loss of smell (ageusia)

Conjunctivitis

Headache

A rash on skin, or discolouration of fingers or toes.

Congestion or runny nose

Nausea or vomiting

Serious Symptoms:

Difficulty breathing or shortness of breath.

Chest pain or pressure.

Loss of speech or movement.

Then we find out some datasets. We run decision tree and logistic regression machine learning algorithms to find out which features are important along with the visualisation of output, Chapter 3 presents the workflow of the learning algorithms in the desired prediction.

6.1 Machine Learning Algorithms

6.1.1. Binary Logistic Regression Machine Learning Algorithm

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Some of the examples of classification problems are Email spam or not spam, Online transactions

Fraud or not Fraud, Tumour Malignant or Benign. Logistic regression transforms its output using the logistic sigmoid function to return a probability value. It is a predictive analysis algorithm and based on the concept of probability. Logistic Regression a Linear Regression model but the Logistic Regression uses a more complex cost function, this cost function can be defined as the ‘**Sigmoid function**’ or also known as the ‘logistic function’ instead of a linear function.

6.1.2 Decision Tree Machine Learning Algorithm

Decision Tree algorithm belongs to the family of supervised learning algorithms. Unlike other supervised learning algorithms, the decision tree algorithm can be used for solving regression and classification problems too. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record’s attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.

6.2 Workflow

6.2.1 Dataset Selection:

The dataset contains the symptoms which can be used to classify the risk of Covid-19 in a person. In the dataset

For the Symptoms:

0->Not present or negative

1->Present or positive

```
Breathing Problem Fever  ... Sanitization from Market COVID-19
0                Yes    Yes  ...                        No       Yes
1                Yes    Yes  ...                        No       Yes
2                Yes    Yes  ...                        No       Yes
3                Yes    Yes  ...                        No       Yes
4                Yes    Yes  ...                        No       Yes

[5 rows x 21 columns]
```

Dataset

Features of the dataset:

1. Breathing Problem,
2. Fever,
3. Dry Cough,
4. Sore throat,
5. Running Nose,
6. Asthma,
7. Chronic Lung Disease,
8. Headache,
9. Heart Disease,
10. Diabetes,
11. Hyper Tension,
12. Fatigue,
13. Gastrointestinal,
14. Abroad travel,
15. Contact with COVID Patient,
16. Attended Large Gathering,
17. Visited Public Exposed Places,

```
18.    Family working in Public Exposed Places,
19.    Wearing Masks,
20.    Sanitization from Market,
21.    COVID-19.
```

6.2.2 Data Cleaning

Here all the entries of the columns are given in yes/no format. We need to change them in 1/0 format.

Output ;

```
Breathing Problem  Fever  ...  Sanitization from Market  COVID-19
0                  1      1  ...                       0          1
1                  1      1  ...                       0          1
2                  1      1  ...                       0          1
3                  1      1  ...                       0          1
4                  1      1  ...                       0          1

[5 rows x 21 columns]
```

6.2.3 Visualisation of data column

By visualising the ‘COVID-19” column data density can be checked.

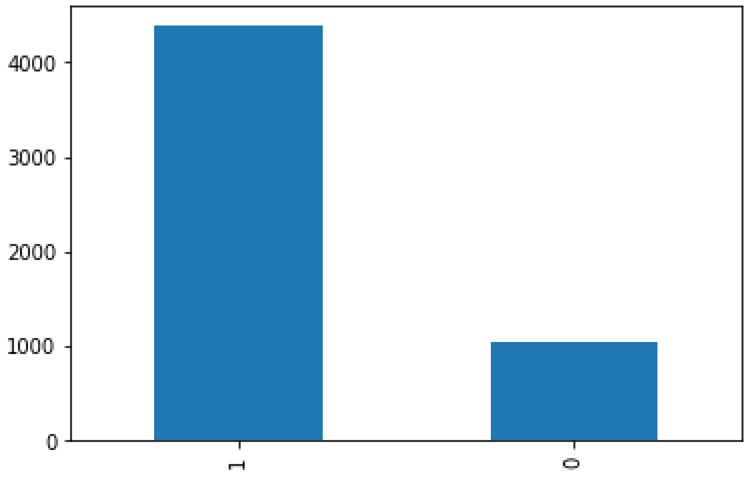


Figure 1 : Visualisation of output column

6.2.4. Application of Decision Tree:

x is the input data (which contains all the symptoms) and y is the output data (which contains the result of prediction). The data needs to be shuffled and splitted to train and test. 80% of data is used for training the model and 20% of it is used to test the performance of our model.

Observation 1 : max_depth = None

```
Decision Tree diagram:
```

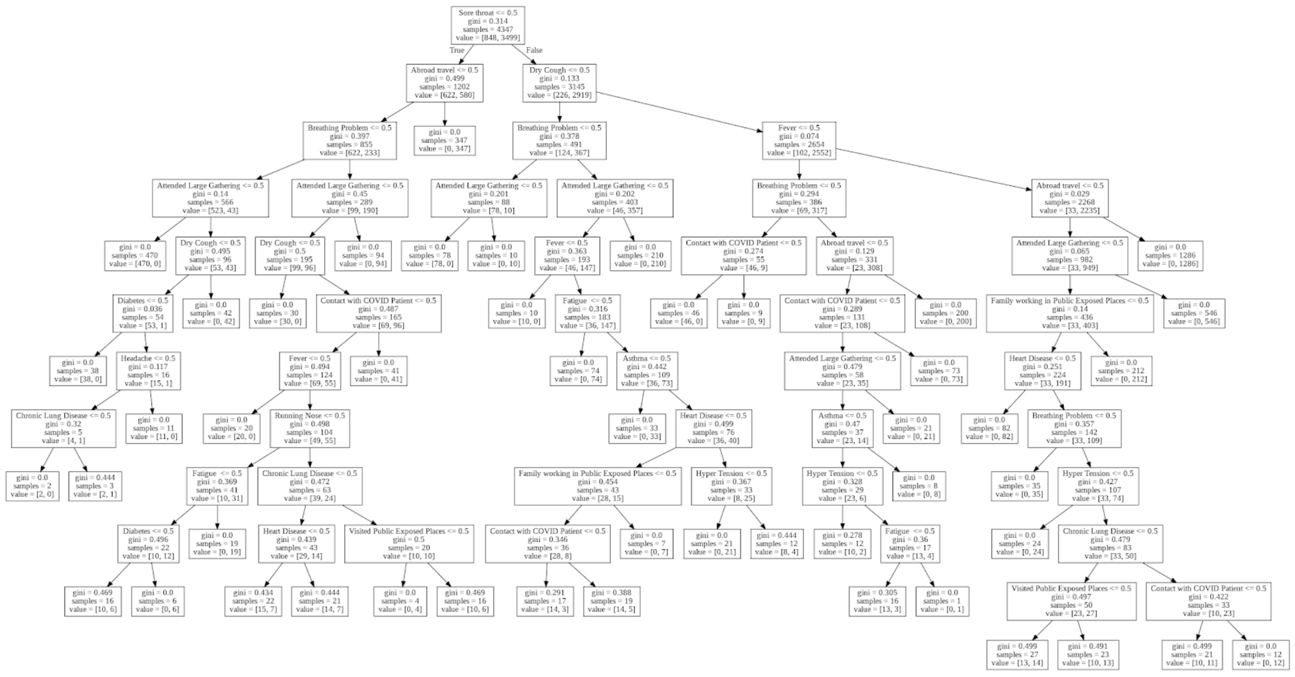


Figure 2 : Decision tree diagram for max_depth = None

```
Accuracy Score on test data :  0.984360625574977
```

Observation from output :

max_depth = None

All features are used except 'Wearing Masks', 'Sanitization from Market'.

Observation 2 : max_depth = 5

```
Decision Tree diagram:
```

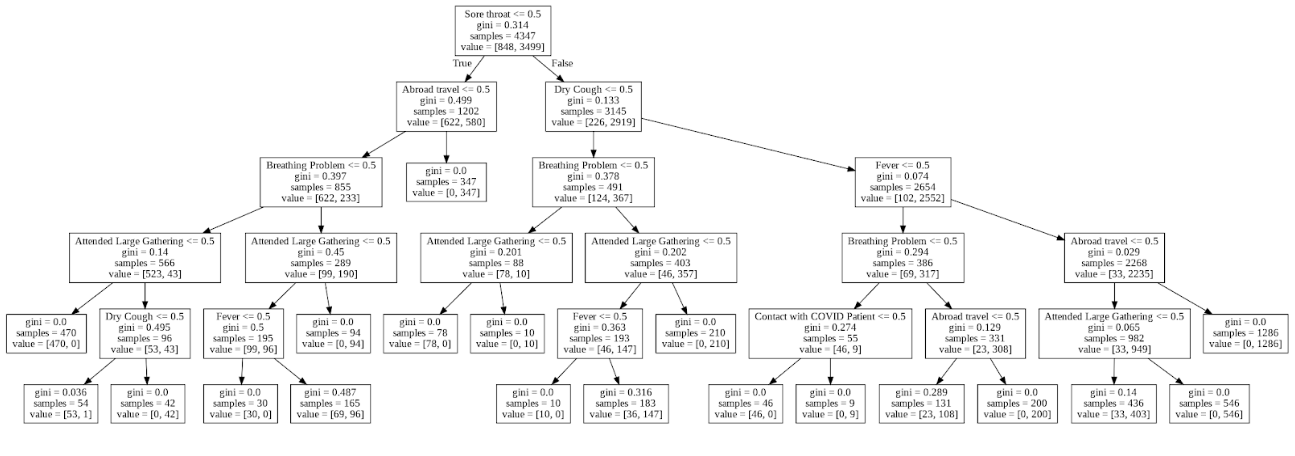

Sore throat <= 0.5 gini = 0.314 samples = 4347 value = [848, 3499]
True
False
Abroad travel <= 0.5 gini = 0.499 samples = 1202 value = [622, 580]
Dry Cough <= 0.5 gini = 0.133 samples = 3145 value = [226, 2919]
Breathing Problem <= 0.5 gini = 0.397 samples = 855 value = [622, 233]
gini = 0.0 samples = 347 value = [0, 347]
Breathing Problem <= 0.5 gini = 0.378 samples = 491 value = [124, 367]
Fever <= 0.5 gini = 0.074 samples = 2654 value = [102, 2552]
Attended Large Gathering <= 0.5 gini = 0.14 samples = 566 value = [523, 43]
Attended Large Gathering <= 0.5 gini = 0.45 samples = 289 value = [99, 190]
Attended Large Gathering <= 0.5 gini = 0.201 samples = 88 value = [78, 10]
Attended Large Gathering <= 0.5 gini = 0.202 samples = 403 value = [46, 357]
Breathing Problem <= 0.5 gini = 0.294 samples = 386 value = [69, 317]
Abroad travel <= 0.5 gini = 0.029 samples = 2268 value = [33, 2235]
gini = 0.0 samples = 470 value = [470, 0]
Dry Cough <= 0.5 gini = 0.495 samples = 96 value = [53, 43]
Fever <= 0.5 gini = 0.5 samples = 195 value = [99, 96]
gini = 0.0 samples = 94 value = [0, 94]
gini = 0.0 samples = 78 value = [78, 0]
gini = 0.0 samples = 10 value = [0, 10]
Fever <= 0.5 gini = 0.363 samples = 193 value = [46, 147]
gini = 0.0 samples = 210 value = [0, 210]
Contact with COVID Patient <= 0.5 gini = 0.274 samples = 55 value = [46, 9]
Abroad travel <= 0.5 gini = 0.129 samples = 331 value = [23, 308]
Attended Large Gathering <= 0.5 gini = 0.065 samples = 982 value = [33, 949]
gini = 0.0 samples = 1286 value = [0, 1286]
gini = 0.036 samples = 54 value = [53, 1]
gini = 0.0 samples = 42 value = [0, 42]
gini = 0.0 samples = 30 value = [30, 0]
gini = 0.487 samples = 165 value = [69, 96]
gini = 0.0 samples = 10 value = [10, 0]
gini = 0.316 samples = 183 value = [36, 147]
gini = 0.0 samples = 46 value = [46, 0]
gini = 0.0 samples = 9 value = [0, 9]
gini = 0.289 samples = 131 value = [23, 108]
gini = 0.0 samples = 200 value = [0, 200]
gini = 0.14 samples = 436 value = [33, 403]
gini = 0.0 samples = 546 value = [0, 546]

Figure 3: Decision tree diagram for max_depth = 5

```
Accuracy Score on test data :  0.9724011039558418
```

Observation from output :

max_depth = 5

Breathing Problem ,Fever ,Dry Cough,Sore throat,Abroad travel,Contact with COVID Patient,Attended Large Gathering.

Observation 3 : max_depth = 4

```
Decision Tree diagram:
```

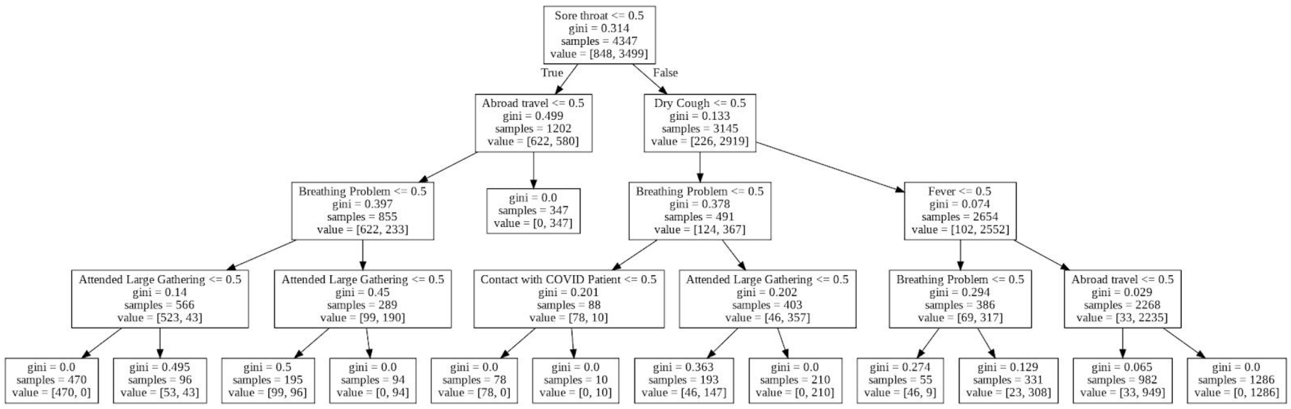


Figure 4 : Decision tree diagram for max_depth = 4

```
Accuracy Score on test data :  0.9558417663293468
```

Observation from output :

max_depth = 4

Breathing Problem ,Fever ,Dry Cough,Sore throat,Abroad travel,Contact with COVID Patient,Attended Large Gathering.

Observation 4 : max_depth = 3

```
Decision Tree diagram:
```

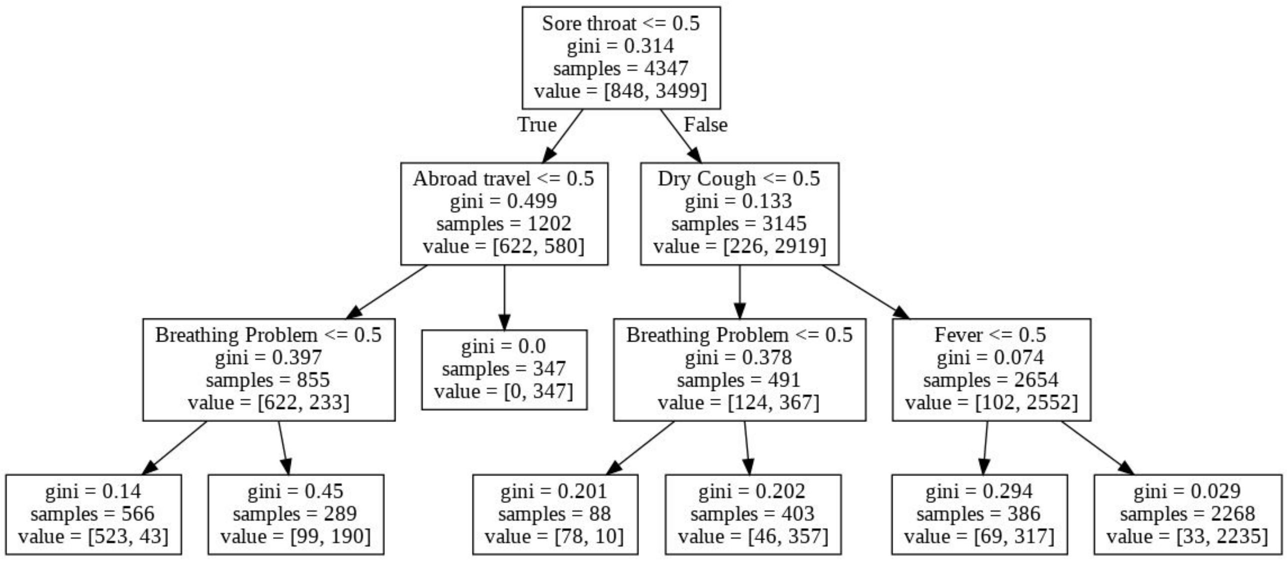


Figure 5 : Decision tree diagram for max_depth = 3

```
Accuracy Score on test data :  0.9337626494940202
```

Observation from output :

max_depth = 3

Breathing Problem ,Fever ,Dry Cough,Sore throat,Abroad travel.

Observation 5 : max_depth = 2

```
Decision Tree diagram:
```

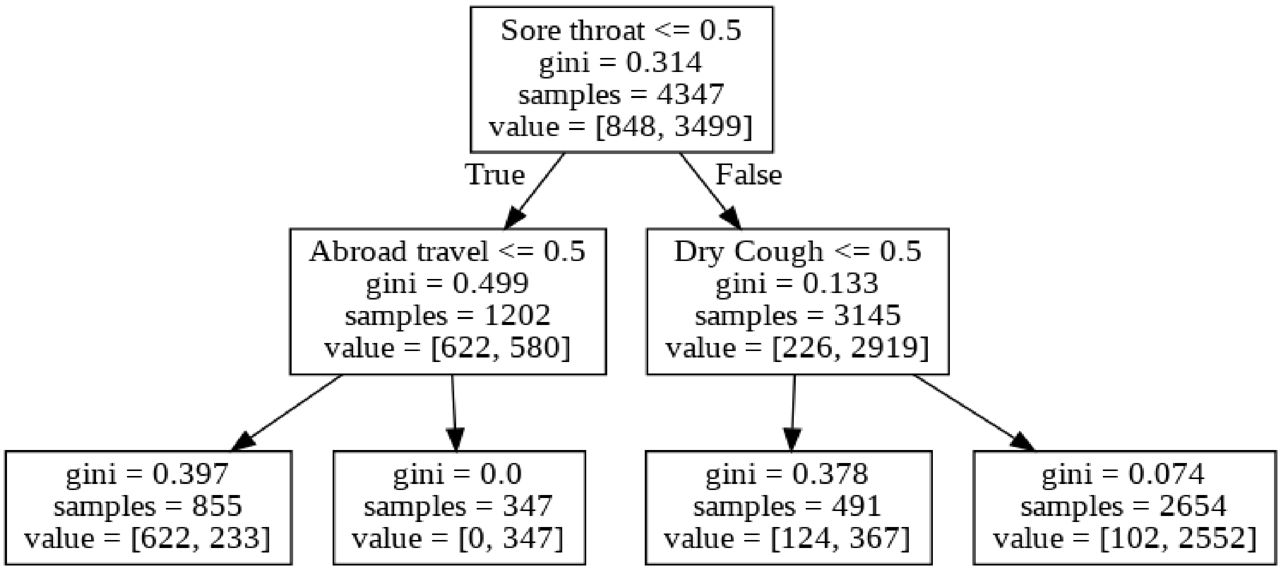


Figure 6 : Decision tree diagram for max_depth = 2

```
Accuracy Score on test data :  0.890524379024839
```

Observation from output :

max_depth = 2

Dry Cough,Sore throat,Abroad travel.

6.2.5 Features based on decision tree output:

Based on the output of decision tree , we can take Breathing Problem ,Fever ,Dry Cough,Sore throat,Abroad travel,Contact with COVID Patient,Attended Large Gathering - these features under consideration for max_depth 5.The reasons for taking these features are

1. If we use None as a parameter in max_depth , we are getting 98% accuracy along with all the 18 features.Measuring all these 18 features with correctness is a difficult task and very error prone.
2. If we use other lower digit as parameter of max_depth then the loss of features and reduction in accuracy is taking a significant role.

6.2.6 Application of Logistic Regression:

x is the input data (which contains all the symptoms) and y is the output data (which contains the result of prediction). The data needs to be shuffled and splitted to train and test. 80% of data is used for training the model and 20% of it is used to test the performance of our model.

6.2.7 Visualisation of output using ROC curve

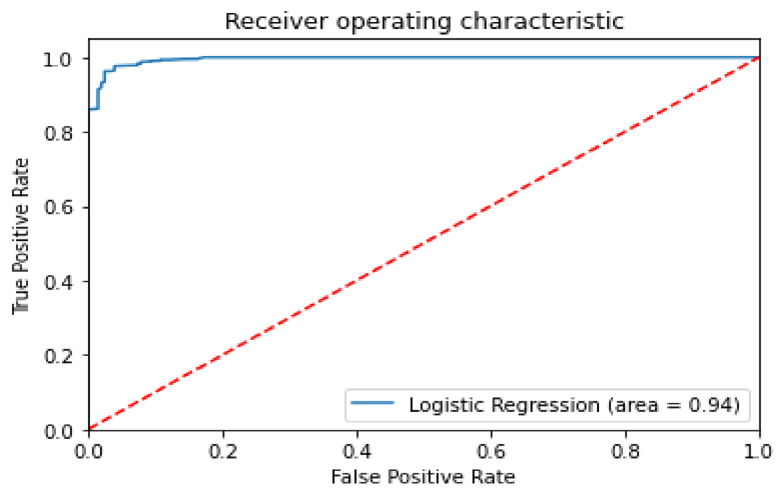


Figure 7 : Receiver operating characteristic curve

6.2.8 Feature selection based on feature dropping

We can find important features by dropping features one by one and observing the change in the accuracy.

Dropped Feature	Accuracy
Breathing problem	96.22 %
Fever	96.59%
Dry Cough	95.67%
Sore throat	96.13%
Running Nose	97.88%
Asthma	97.245
Chronic Lung Disease	97.14%
Headache	97.33%
Heart Disease	97.42%
Diabetes	97.42%
Hyper Tension	97.24%
Fatigue	97.33%
Gastrointestinal	97.42%
Abroad Travel	95.58%
contact with COVID-19 patient	96.87%
Attend Large Gathering	95.40%
Visited Public Exposed Places	97.42%
Family working in public Exposed places	97.33%
Wearing Mask	97.24%
Sanitization from Market	97.24%

Observation from output:

Accuracy on test dataset with all features = 97%

The features when dropped accuracy decreased by 96%:

Breathing Problem, Fever, Sore Throat, Contact with COVID Patient.

The features when dropped accuracy decreased by 95%:

Dry Cough, Aboard Travel, Attended Large Gathering.

6.2.9 Final experiment on dataset using the selected features

```
x=df.drop(columns=['Running Nose', 'Asthma', 'Chronic Lung Disease',
'Headache',
       'Heart Disease', 'Diabetes', 'Hyper Tension', 'Fatigue ',
       'Gastrointestinal ', 'Visited Public Exposed Places',
       'Family working in Public Exposed Places', 'Wearing Masks',
       'Sanitization from Market','COVID-19'])
y=df['COVID-19']
fun(x,y)
```

```
Index(['Breathing Problem', 'Fever', 'Dry Cough', 'Sore throat',
       'Abroad travel', 'Contact with COVID Patient',
       'Attended Large Gathering'],
      dtype='object')
Confusion Matrix :
 [[183  20]
 [ 17 867]]

Accuracy Score on train data: :  0.9650333563377042

Accuracy Score on test data :  0.9659613615455381
```

6.2.10 Visualisation of output using ROC curve

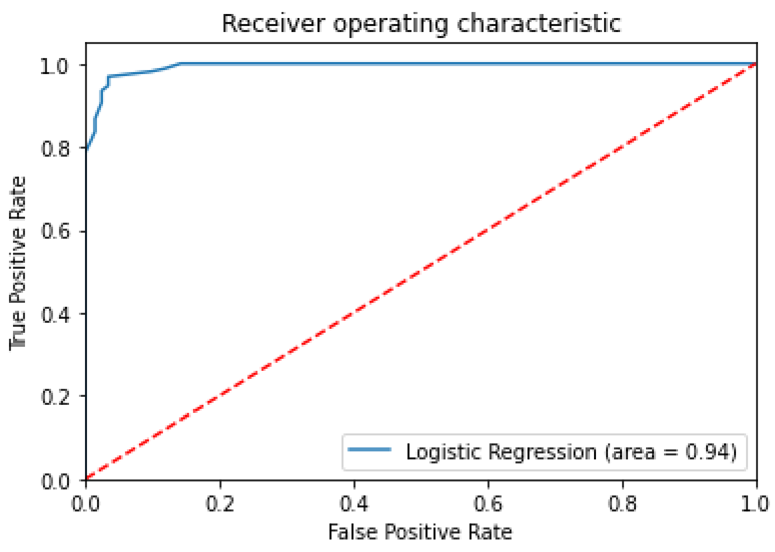
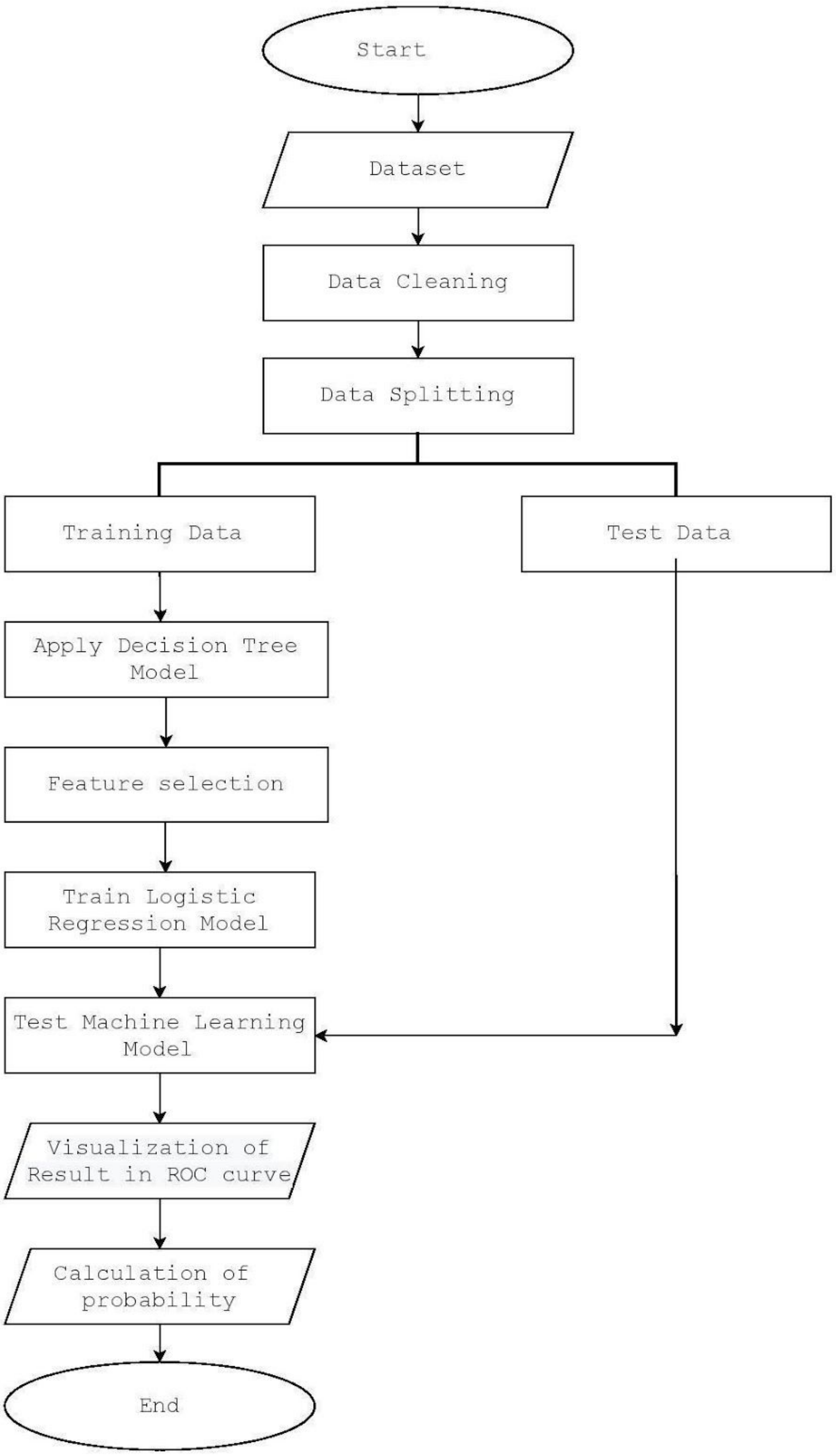


Figure 8 : Receiver operating characteristic curve

6.3 Flowchart



Start
Dataset
Data Cleaning
Data Splitting
Training Data
Test Data
Apply Decision Tree Model
Feature selection
Train Logistic Regression Model
Test Machine Learning Model
Visualization of Result in ROC curve
Calculation of probability
End

7. Input and output from project Contactless Health Monitoring System

7.1. Overview of the project:

Contactless Health Monitoring System is a IOT based Monitoring system using sensors to avoid direct contact with patients. The sensors it uses are

1. Temperature sensor
2. Pulse rate sensor
3. Humidity sensor
4. Room temperature sensor
5. Smoke sensor.

We can use a temperature sensor to sense the body temperature and based on the threshold fever can be detected. This can be used as an input in the Machine Learning Model.

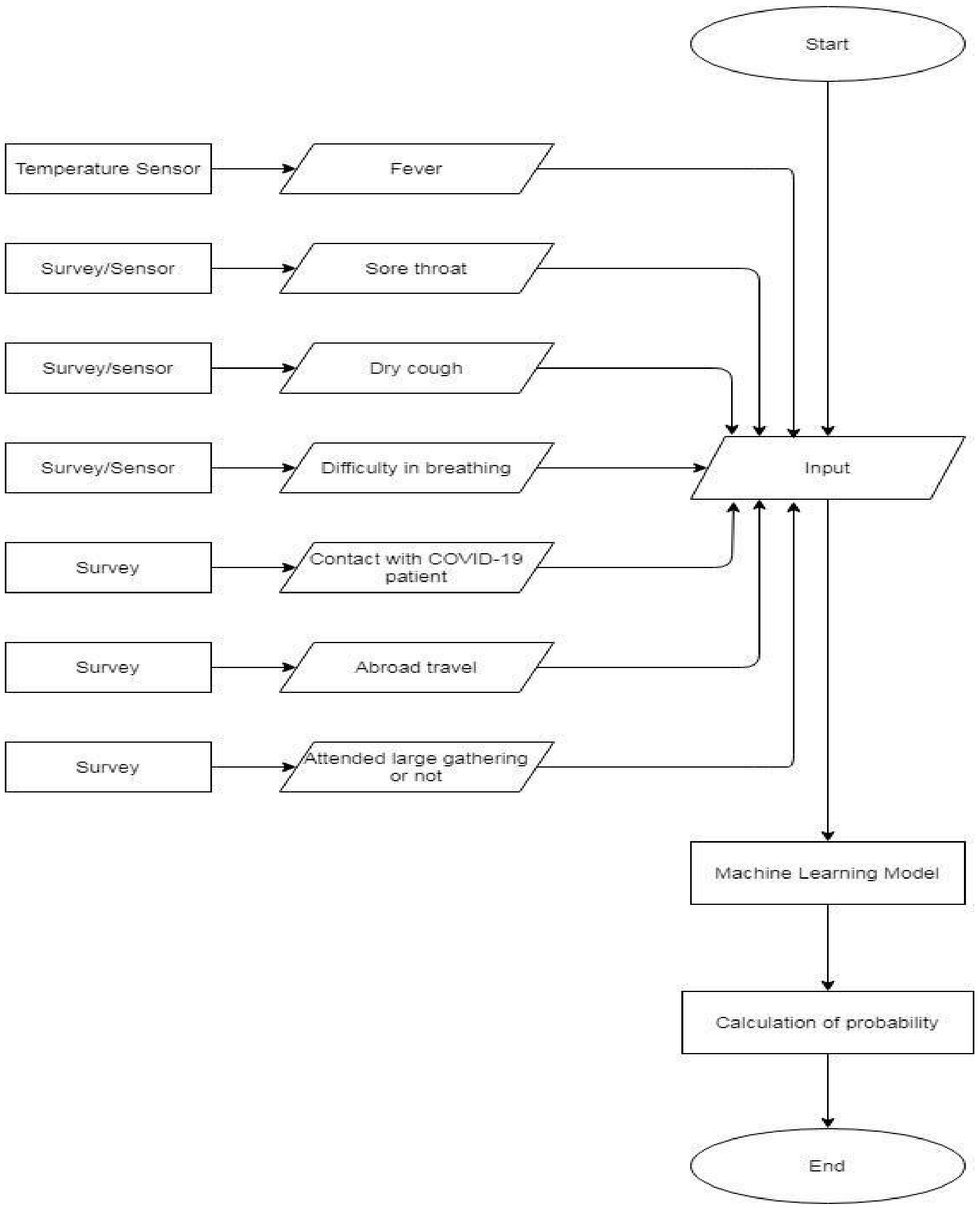
7.2. Benefits of using the project;

Instead of using only survey-based prediction, sensor-based prediction is more accurate and error free because in case of survey-based analysis there may be some human error in the input given by the user which may cause wrong prediction.
Here we are using both sensor and survey-based analysis to get optimal result. Survey can be used in some questions like

1. Attended large gathering or not,
2. Is there any contact with COVID-19 patient?
3. If the user has any Abroad travel history.

Whereas some questions like fever, sore throat, dry cough survey may not give the correct input so we can measure temperature using the Temperature Sensor then we can feed the input to the Machine Learning Model.

7.3. Flowchart showing the overall communication:



8. Conclusion

To revive the tourism sector this project can be used as a hallmark Idea. Hear the web application can be used as single window solution for the traveller. New hotel be it as quarantine/non-quarantine centre can be registered very easily and all these stuffs can be swiftly managed by Admit. Again, amid pandemic which requires robust innumerable health infrastructure, IoT can be used to reduce the man power and manage the health parameters with effectively and efficiently.

In COVID-19 possibility detector Logistic Regression Machine Learning Algorithm is used to calculate the probability of getting infected using the symptoms. This predictor will be helpful in a time which requires innumerable testing though resources are less so selective testing can be done.

This project can be very helpful not only to boost travel sector but also to bring livelihood back.

9. Future scope

- Payment method can be fully online in the hotel management web application.
- IoT based Remote Patient Monitoring System can be enhanced to detect and collect data of several anomalies for monitoring purpose such as Blood pressure, home ultrasound, Brain signal monitoring, Tumor detection etc.
- The interface can be designed to control which sensors can be used by consumers according to their needs.
- Web UI can be enhanced to perform several activities which include controlling the hardware, real-time graphs, history and analysis graphs to observe anomalies etc.
- COVID-19 is one of the major issues in today's world. The infection rate is also rising very rapidly. The frontlines are in a great risk as they are working in close contact with COVID-19. We can use different sensors and another advanced machine learning algorithm to detect breathing difficulty, sore throat, dry cough, runny nose, body pain to accurately measure and treat COVID-19 infected patient with the help of technology. It will not only reduce infection rate but also can reduce cost of treatment with the help of this features we can accelerate COVID-19 free world.

10. References

http://www.w3schools.com/

https://www.javapoint.com/

http://www.arduino.org/

http://www.wikipedia.com/

Symptoms resources :

https://www.who.int/health-topics/coronavirus

https://www.icmr.gov.in/cfaqs.html

https://www.cdc.gov/coronavirus/2019-ncov/index.html

https://m.tribuneindia.com/news/nation/icmr-expands-coronavirus-symptoms-list-includes-loss-of-taste-and-smell-muscle-pain-diarrhoea-98665

Logistic regression advantage : https://machinelearning-blog.com/2018/04/23/logistic-regression-101

Python advantage : https://steelkiwi.com/blog/python-for-ai-and-machine-learning

Logistic Regression Machine Learning Algorithm :

https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148

Decision Tree Machine Learning Algorithm ;

https://towardsdatascience.com/decision-tree-algorithm-explained-83beb6e78ef4

Dataset :

https://www.kaggle.com/hemanthhari/symptoms-and-covid-presence

Confusion Matrix :

https://towardsdatascience.com/demystifying-confusion-matrix-confusion-9e82201592fd

ROC curve :

https://en.m.wikipedia.org/wiki/Receiver_operating_characteristic

Probability :

https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html#sklearn.linear_model.LogisticRegression.predict_proba

Project Report Theory :

http://www.iitb.ac.in/newacadhome/rules/Dissertation17june09-10.pdf

http://gssst.iitkgp.ac.in/uploads/student69/Varun%20Bedi_Thesis.pdf

http://web.iitd.ac.in/~sankalpa/nupurthesis.pdf