PROJECT REPORT

INTRODUCTION

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Smart Solutions For Railways is to manage Indian Railways is the largest railway network in Asia and additionally world's second largest network operated underneath a single management. Due to its large size it is difficult to monitor the cracks in tracks manually. This paper deals with this problem and detects cracks in tracks with the help of ultrasonic sensor attached to moving assembly with help of stepper motor. Ultrasonic sensor allows the device to moves back and forth across the track and if there is any fault, it gives information to the cloud server through which railway department is informed on time about cracks and many lives can be saved. This is the application of IoT, due to this it is cost effective system. This effective methodology of continuous observation and assessment of rail tracks might facilitate to stop accidents. This methodology endlessly monitors the rail stress, evaluate the results and provide the rail break alerts such as potential buckling conditions, bending of rails and wheel impact load detection to the concerned authorities.

1.2. PURPOSE

Internet is basically system of interconnected computers through network. But now its use is changing with changing world and it is not just confined to emails or web browsing. Today's internet also deals with embedded sensors and has led to development of smart homes, smart rural area, e-health care's etc. and this introduced the concept of IoT . Internet of Things refers to interconnection or communication between two or more devices without human to-human and human-to-computer interaction. Connected devices are equipped with sensors or actuators perceive their surroundings. IOT has four major components which include sensing the device, accessing the device, processing the information of the device, and provides application and services. In addition to this it also provides security and privacy of data . Automation has affected every aspect of our daily lives. More improvements are being introduced in almost all fields to reduce human effort and save time. Thinking of the same is trying to introduce automation in the field of track testing. Railroad track is an integral part of any company's asset base, since it provides them with the necessary business functionality. Problems that occur due to problems in railroads need to be overcome. The latest method used by the Indian railroad is the tracking of the train track which requires a lot of manpower and is time-consuming

LITERATURE SURVEY

2.LITERATURE SURVEY

2.1 EXISTING SYSTEM

[1]Journal Name:. “Integrating automatic verification of safety requirements in railway interlocking system design”, The 6th IEEE International Symposium on High Assurance Systems Engineering (HASE’01), Washington, USA 2011.

Author Name:POOVIZHI S

The sensors are made up of a transistor, an Op-amp, a handful of resistors, and a few IR leds. A wireless sensor network (WSN) is a network of autonomous sensors-equipped devices that is spatially distributed and wireless. This WSN technology offers distributed nodes and wireless communication to the wired world. The 900 MHz frequency used by the wireless protocol is chosen based on the needs of the application. The protocol uses 2.4 GHz radios that are compliant with IEEE 802.15.4 or IEEE 802.11 (Wi-Fi) standards. The issues that train passengers encounter are numerous. One of them is the absence of water in the train, travellers taking long distance trains must either travel with a meagre supply of water or without any at all. The availability of seats in trains is another issue. To purchase tickets for the train they want to take, passengers must wait in line for a very long time. It will be quite difficult for a passenger to go by train if there is nowhere for them to sit. For operation, the IR module uses 358 comparator ICs. When it detects an IR frequency, the sensor's output changes to logic 1, otherwise to logic 0. Leds can be used to examine the sensor's state, and no further hardware is needed.This means there are no open seats for incoming guests. The availability of seats for new customers is indicated if the IR led did not detect any reflected signal. Normally, the output pin is low.The receiver LED will be off even though the IR LED is continuously transmitting since there is nothing to reflect back to the IR receiver owing to an obstruction. The IR receiver's output decreases when an obstruction is encountered. The obstacle surface reflects the IR signal. Thecomparator's

output will be driven low as a result. The LED's cathode is then linked to this output, which causes it to illuminate.

[2]Journal Name:. “Real-time rail head surface defect detection: A geometrical approach,” in Proc. IEEE Int. Symp Indust. Electron., 2009.

Author Name: Pranav

Using an ultrasound testing approach to find the rail track's defects using an ultrasonic distance metre. When a crack is found, the appropriate coordinates are transmitted to the nearby station. The GPS and GSM module are used to record and transfer the coordinates. The best way for locating small cracks and determining their rate of expansion is the ultrasonic technique. At regular periods, the growth rate can be observed. A fracture detection non-destructive system is used. Non-destructive testing technique is one of the procedures that aid in the inspection of material without doing any damage. NDT is a popular technique for maintaining materials without addressing the fundamentals of the material. NDT is a popular technique for maintaining materials without addressing the fundamentals of the material. Due to the diverse behaviours that ultrasonic waves exhibit in various material qualities, they are heavily utilised in this technology. ultrasonography is used. Every area of the permanent way is examined every day on foot. Gang patrol during unusual rainfall, night patrol during the monsoon, hot weather patrol for welded track, security patrol, watchmen at susceptible areas, and cold weather patrol are some of the patrolling types. Gang patrol during rain should have an effect on the length, which should be affected. It operates apart from other patrolling. The meteorological department sends out telegrams to warn people about storms and heavy rain. Watchmen and Gang members are on high alert and ready to start patrolling. Security patrols are conducted to safeguard trains from track tampering and obstructions on the route, as well as to find rail track faults using an ultrasonic testing technology. When a crack is found, the appropriate coordinates are transmitted to the nearby station. The GPS and GSM module records and transmits the coordinates. The best system is the ultrasonic approach, which

can even find little cracks and estimate how quickly they will spread. Following multiple measurements made at regular intervals, the growth rate can be determined. Non-destructive testing technique is one of the processes that aid in material evaluation without doing any damage. NDT is a popular technique for maintaining materials without addressing the fundamentals of the material. Because ultrasonic waves exhibit a variety of behaviours in different material characteristics, they are often used in this procedure. When an ultrasound wave signal travels from one distinct medium to another, some of the signal energy travels over to the other medium while the remaining energy is reflected back.

[3]Journal Name:. "Safety verification for train traffic control communications", IEEE journal on selected areas in communications, vol. Sac4, no. I, 2012

Author Name: Bharti.S.Dhande ,Utkarsha S.Pacharaney

The most common level crossing controllers and train tracks to use IR sensors and the internet of things In India, the means of transportation is widely employed. It is a form of transportation that encounters a any difficulties brought on by human mistakes, like level cross collisions, broken-down vehicle collisions follow etc. a level crossing or a road intersection a railway line ,which calls for human coordination, the absence of which results in accidents, as well as the A primary issue with railroad analysis is detection. the position of the crack. If this issue is if not contained at an early level, they could a lot of derailments with significant loss of life life and possessions. In the conventional system, the gatekeeper is responsible for controlling level crossings. The gatekeeper receives instructions from the control room via telephone at the majority of the level crossings. However, the likelihood of manual error at these level crossings is considerable and risky without actual knowledge of the train schedule. Accidents on the railroad could result from delayed gate opening and shutting. The concept of railway gate automation and crack detection system has been modified by employing IR sensors and IOT technology, which performs

automatic gate operation and aids in identifying broken track, in order to eliminate human errors during the operation of gates and derailment. In this system proposal, an LPC2148 microcontroller was used.It is a small microprocessor with low power requirements. LPC2148 are perfect for applications where downsizing is a major need, such as access control systems, because to its small size and low power consumption. It has numerous UARTs, SPI, SSP, and I2C serial communication interfaces in addition to a USB 2.0 Full Speed device. It has 8 kb to 40 kb of on-chip SRAM. Devices are therefore excellent candidates for communication gateways. In this paper, we make a suggestion. Before beginning the rail-way line scan as part of the crack detection system, the robot is programmed to self-calibrate the IR transmitter and receiver. The robot must wait a certain amount of time after calibration for the GPS module to begin reading the correct geographic coordinate. The idea behind this crack detection is that the amount of light that reaches the IR receiver is inversely proportionate to the crack's intensity. The IR transmitter and receiver will be mounted on the rail in a straight line. When the transmitter's light does not hit the receiver during operation, the device does not detect a crack. And when the receiver receives light from the transmitter. We employed a GPS receiver, whose purpose is to obtain the most recent latitude and longitude information, so order to determine the train's current location in the event of crack detection.

[4]Journal Name:. Safety verification for train traffic control communications

Author Name: G.Tarnai

In this study, it is suggested that RFID-based chip cards be read and scanned at a distance using a technique called distance readability. Potential free riders can be effectively caught using the distance reading. Distance scanning by itself will be unable to ascertain the precise number of free riders, but a second technique to count the population of an area is recommended. This research proposes a method to identify free riders early on based on the insight of merging the two technologies (RFID distance scanning and People

thermal image counting). This paper's focus will be on the structure and architecture required to record Faredodger's study, which will be put to use to run tests in an experiment to confirm the presumptions.

[5]Journal Name: .Autonomous railtrack inspectionusing vision based system," in Proc. IEEE Int. Conf. Comput. Intell. Homeland Secur. Pers. Safety, 2009

Author Name: Smita S. Bhavsar

RFID method to prevent aircraft collision the railway transportation network is thought to be the safest and simplest network, however it is no longer that much safer since numerous crashes and accidents happen due to poor network communication, incorrect signalling, bad weather, and sudden changes in track or route. Due to the speed of moving trains, which necessitates a lead space for stopping, it is exceedingly challenging to prevent such collisions. Around the world, there have been several train accidents. According to a CNN IBN India story dated September 2011 Human mistake accounts for 85% of train accidents, either the driver or the main control room before a collision. There is currently no way to prevent train collisions. ACD (anti-collision device) system-based solutions have been put into place by Indian Railways. Due to their design concept of using GPS for track recognition and having a high implementation cost, they have inherent issues in the Station portion and close to mountains. My system, which relies on RFID, ARM Controller, and GSM to assist solve the aforementioned issues, uses automated surveillance to help eliminate train accidents. Each train reads and transmits its track id to surrounding trains in this system, which assigns a track id to each train track. if there are two trains travelling at the same time.

[6]Journal Name:. Crack Detection System For Railway Track By Using Ultrasonic And Pir Sensor" IJAIC-2014

Author Name: Shiladitya Ghosh, PallabDasgupta, Chittaranjan Mandal, AlokKatiyar

The authenticity of the movement authorities provided by the control

centre will have a big impact on the automatic train controller system. A Radio Block Centre (RBC) in the European Train Control System (ETCS) is in charge of issuing movement permits to all trains that are under its control in a fashion that ensures the train's safe movement. In ERTMS/ETCS Level-1, the RBC receives train position data via train detection equipment; however, in Levels 2 and 3, the train itself uses its on-board radio to transmit its position. Obtaining formal proof that the method for granting movement authorization is safe is necessary due to the rising complexity of train movements across locations, which necessitate greatly variable speed profiles at various times in time. The core of this framework is a verification engine that demonstrates that, given an inertial model of the train, the RBC's movement authorizations guarantee that the movements of the trains satisfy all restrictions. The presented model does not take into account every component of the total ETCS system. European Railway Traffic Management System is referred to as ERTMS. For the two trains that we have taken into account as being a component of the system, we define two distinct models. The basic design of both trains is the same. A crucial step in assuring the general security of automatic train control systems is the formal verification of the movement authorities provided by the track-side radio control block (RBC).

[7] Journal Name: "Solid-state interlocking(SSI): an integrated electronic signaling system for mainline railways", IEE proceedings, 2012.

Author Name: A. .H. Cribbens

In the fast developing country, people are facing many accidents; it would be indesirable for any nation to losing their life for unwanted cause. Railways are one of the important transports in India. There is a need for manual checking to detect the crack on railway track and always railway personnel takes care of this issue, even though the inspection is made regularly. Sometimes the crack may unnoticed . Because of this the train accident or derailment may occur. In order to avoid this situation and

automate therailway crack detection has been proposed. Here ultrasonic sensor is used to detect the crack in the railway track by measuring distance from track to sensor, if the distance is greater than the assigned value the microcontroller identifies there is a crack, also it tells the exact location of the crack by the formula "DISTANCE=SPEED*TIME". While the checking process is going on, the train may approach, it is identified by the vibration sensor and gives alert to the microcontroller, there by shrinks the size of the robot between the two tracks. After the train has crossed it returns to its normal position and continue its checking process.

[8]Journal Name: "Characterisation of defects in the railhead using ultrasonic surface waves," NDT & E Int., vol. 39,no. 6, pp. 468–475, 2006.

Author Name: R. Edwards, S. Dixon, and X. Jian.

The Indian Railways has one of the largest Railway networks in the world, crises- crossing over 1,15,000 km in distance, all over India. However, with regard to reliability and passenger safety Indian Railways is not up to global standards. Among other factors, cracks developed on the rails due to absence of timely detection and the associated maintenance pose serious questions on the security of operation of rail transport. A recent study revealed that over 25% of the track length is in need of replacement due to the development of cracks on it. Manual detection of tracks is cumbersome and not fully effective owing to much time consumption and requirement of skilled technicians. This project work is aimed towards addressing the issue by developing an automatic railway track crack etection system integrating an infrared red (IR) crack sensing module and a communication module based on GSM technology by which information about the location of the crack can be conveyed to a central location enabling the immediate attention and intervention of maintenance personals.

[9]Journal Name: "Ultrasonic characterisation of defects in rails," Insight, vol.44, no. 6, pp. 341–347, 2002.

Author Name: R. Clark, S. Singh, and C. Haist.

In India, as most of the commercial transport is carried out through the rail network, problems with this network can be highly damaging to the economy, regardless of the social consequences of loss of life or limb. I have. This white paper proposes an inexpensive yet robust solution to the railway breakage detection problem. The method is simple in idea, but completely new and unique in the sense that it has not been tested to date. This paper describes the technical and design aspects in detail and also provides a proposed robust crack detection algorithm. The paper also presents details of his RRCDS implementation results using simple components such as a GPS module, a GSM modem and an LED-LDR based crack detector assembly. The proposed scheme is modeled for robust implementation in the Indian scenario.

[10]Journal Name: "Development of a machine vision system for inspection of railroad track".

Author Name: S.Sawadisavi , J.Edwards, E.Resend, J.Hart, C.Barkan, and N.Ahuja.

In European cities, the majority of the public transit infrastructure is easily accessible. The majority of the train stations are positioned in an open and "gate-free" environment, easy available to everyone and hence presents possible problems in the system. Due of this, fare dodging boarding a tram or train without purchasing a ticket is simple. This study proposes a conceptual framework and architecture to detect and track passengers using an RFID distance scan in conjunction with people counting methods, with the goal of capturing free riders in an early stage. It is a ticketing system based on RFID that utilises a OV-Chip card is a smartcard. The findings demonstrate that using an alternative system architecture increase in getting free trips inspectors are at a far early stage.

2.2 REFERENCES

[1] POOVIZHI S* Assistant Professor Department of Electronics and Communication Engineering R.M.K.College of Engineering and Technology Puduvuyal, Tamilnadu, India.

[2] Pranav Lad Production and Industrial Engineering VIT University Vellore, India.

[3] Bharti.S.Dhande ,Utkarsha S.Pacharaney Department of Electronics and Telecommunication Engineering DMCE, University of Mumbai, Airoli, Navi Mumbai – 400708.

[4] G.Tarnai, "Safety verification for traintraffic control communications", IEEE journalon selected areas in communications, vol. sac-4,no. I, 2012.

[5]Smita S. Bhavsar Department of E&TC Engineering, Zeal Education Society's Zeal College of Engineering and Research, Maharashtra, Pune, India.

[6] Shiladitya Ghosh, PallabDasgupta, Chittaranjan Mandal, AlokKatiyar Department of Computer Science andEngineering Indian Institute of Technology Kharagpur Research Development & Standards Organization, Indian Railways, Lucknow.

[7] A. .H. Cribbens, "Solid-state interlocking(SSI): an integrated electronic signaling systemfor mainline railways", IEE proceedings, 2012.

[8] R. Edwards, S. Dixon, and X. Jian, "Characterisation of defects in the railhead usingultrasonic surface waves," NDT & E Int., vol. 39,no. 6, pp. 468–475, 2006.

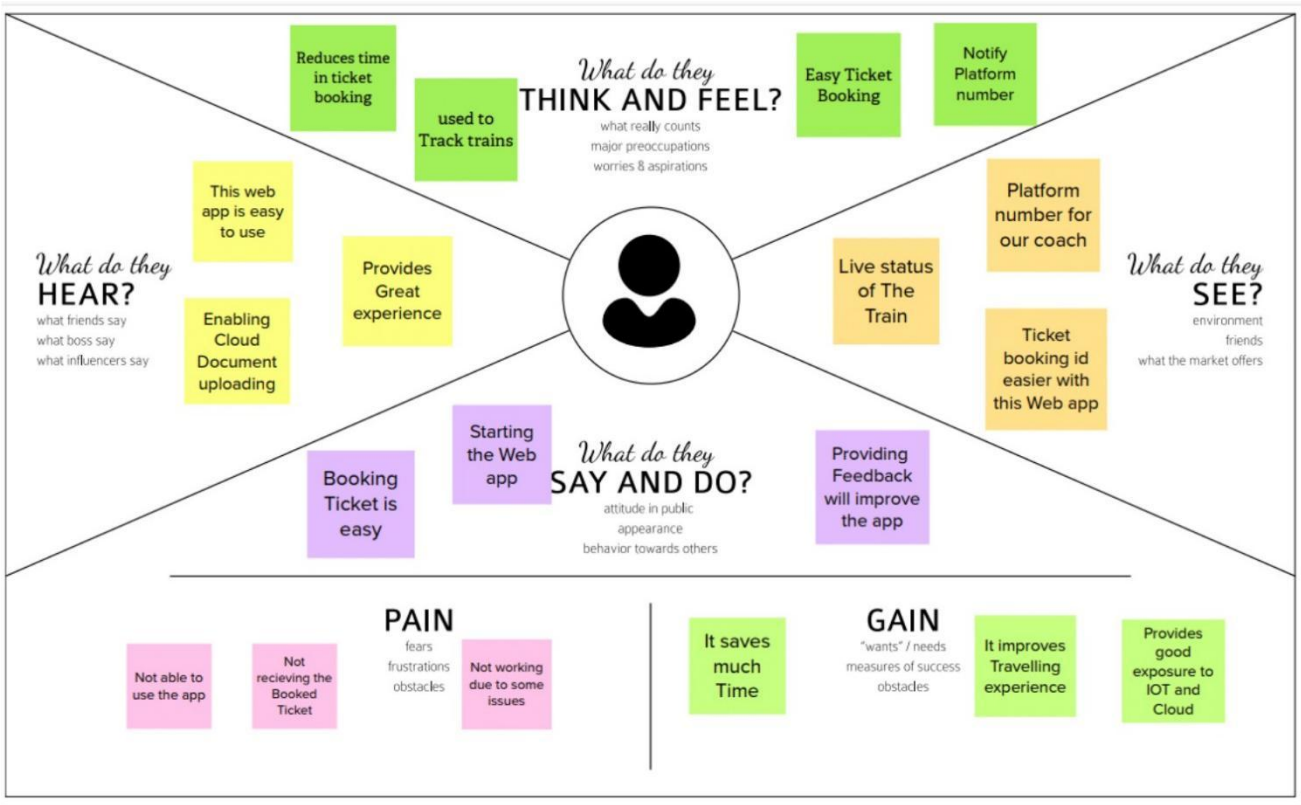
[9] R. Clark, S. Singh, and C. Haist, "Ultrasonic characterisation of defects in rails," Insight, vol.44, no. 6, pp. 341–347, 2002.

[10] S.Sawadisavi J.Edwards, E.Resend, J.Hart, C.Barkan, and N.Ahuja ,"Development of a machine vision system for inspection of railroad track," in Proc. Amer. Railway Eng. Maintenance way Assoc. Annu. 2012.

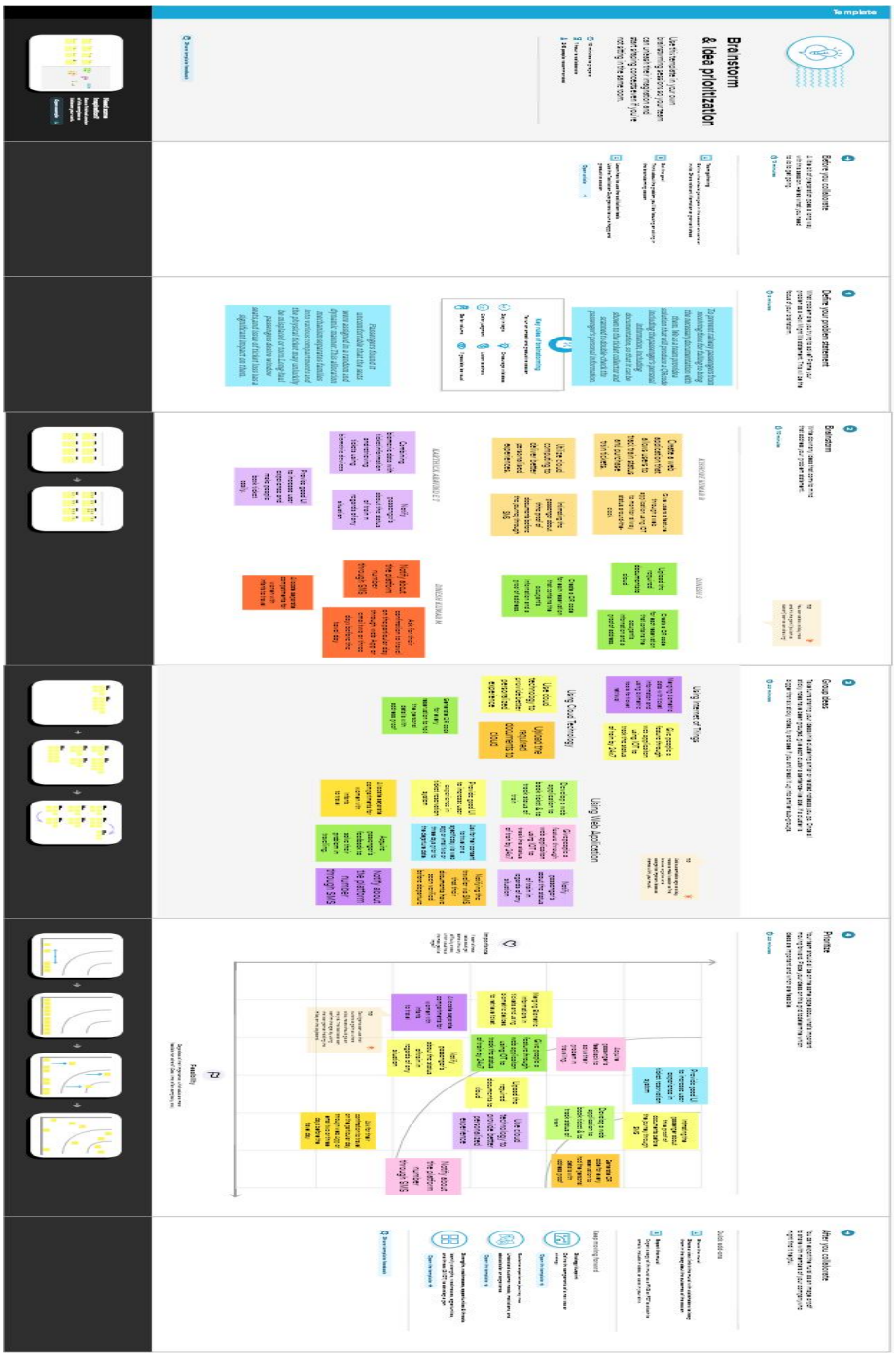
IDEATION AND PROPOSED SOLUTION

3. IDEATION AND PROPOSED SOLUTON

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTORMING



3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	➢ The passenger convenience in making ticket reservations through the counter is poor. ➢ There will be no information about the ticket availability until all the ticket has booked. ➢ The printed tickets may be erased or teared by moisture, which is a problem for the traveller. The usage of paper tickets was to blame for this. ➢ The passengers will encounter the problem of being unable to reserve the preferred seat. ➢ While travelling either with family or friends the seats were distributed randomly. so they can't interact with each other properly as they thought.
2.	Idea / Solution description	➢ The user can book tickets using the website, where they will receive a QR code which can be scanned instead of using tickets to retrieve the user's information. ➢ By installing a GPS module inside the train, website can also display the train's real-time positions. The journey's location will be regularly updated on the website. ➢ Additionally, the website enables users to reserve the desired seat.
3.	Novelty / Uniqueness	➢ The webpage will offer the customer a QR code, which will cut down on

		paperwork. ➢ It allows the user to reserve the preferred seat. ➢ All of the client booking information will be saved in the database with a special ID which can be retrieved when the ticket collector scans the QR Code.
4.	Social Impact / Customer Satisfaction	➢ There is no need going to the station to book tickets because they can be booked online, and the transaction process is also made simple. ➢ All confirmations and cancellations will be sent to the consumer by provided email or mobile phone.
5.	Business Model (Revenue Model)	➢ The user of this application can check the seat availability and they can select the seats to their convenience. ➢ It makes the ticket booking simple for the clients to schedule daily shuttles and journeys, and it eliminates carrying around tickets. The customer can also view the train's current location. ➢ For using the abovementioned facility, a specific amount of fees may be charged, particularly if a customer wants to reserve their preferred seat they must pay extra for an ticket.
6.	Scalability of the Solution	➢ Elimination of physical paper tickets ➢ While booking ticket in counter the clients had to carry cash and while booking E- ticket you are paying through online directly from bank or payment

		apps which makes work more easy for the clients. ➢ This reduces the wastage of the papers and the environment.

3.4 PROBLEM SOLUTION FIT

1. CUSTOMER SEGMENT(S)	6. CUSTOMER CONSTRAINTS	5. AVAILABLE SOLUTIONS
• Functional Traveler • Day Tripper • Tourist • Leisure-Hedonic Traveler • Office goers • College Students • Bloggers & Vloggers	• I am unable to book the window seat. • Missed the Train Ticket. • While Raining the Ticket gets wet and tore. • Seats for Friends were allotted in different Compartments.	• Using the QR Code instead of Physical Paper Tickets. • Providing an Service Through Web Application. • Using biometric verification or cloud technology

2.JOBS-TO-BE-DONE/ PROBLEMS	9. PROBLEMT ROOT CAUSE	7. BEHAVIOUR
• Replacing a QR Code instead of Physical Ticket Papers • Using an Web Application which gives an Option to Select the desired Seats while booking	• Dynamic Allocation of Seat by the Railway Department. • Providing an Physical copy of Tickets to the Passenger. • Carelessness of the Passenger which leads them to loss the Ticket	• The desired seat selection option through web app admires the travelers. • Using The QR code instead of Paper Tickets attract the senior citizen who forgets the things always.

3. TRIGGERS	10. YOUR SOLUTION	8. CHANNELS of BEHAVIOUR
• The advertisement through billboards in Junction triggers the passenger to utilize it. • Reading about a more efficient solution in the news.	• Using the QR Code instead of Physical Paper Tickets. • Giving an Option in a Web application to Select the desired Seats while booking Tickets	• **Online** : They should provide proper login credentials which may helps to get a duplicate copy of E-Ticket • **Offline** : In this, Passenger should carry a copy of Ticket and a QR code
4.EMOTIONS: BEFORE/AFTER		
• Before the Problem they lead an normal travel. • If the Problem arise they may feel insecure, guilty and some may get anger		

REQUIREMENT ANALYSIS

4.REQUIREMENT ANALYSIS

4.1. FUNCTIONAL REQUIREMENTS

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Before the user registration there will be language selector .All the language is applicable .When user enter in to the website they can see the page which shows enter the email-id ,mobile number and name. After that in screen it shows the verification code which will be sent to the email-id.
FR-2	User verification	The verification code is send to the registered email id.
FR-3	User confirmation	The verification code is entered in to the website. After finishing that home page is opened.
FR-4	Process of booking	When the home page is opened there will be a From and To options. We must enter the details then after that we can able to see the number of trains availability and seats availability. We can select the particular train and particular seats which we need and click the confirm option.
FR-5	Payment process	After entering all the details select the payment option like UPI apps , Net-banking , etc., When we select the comfortable method then it process through selected payment option then payment can be done carefully and securely, then the ticket will be confirmed. After confirmation it will return to the page and we can see the details of booking.
FR-6	Confirmation message	After all the process has been completed the QR code will be send to both mobile number(via SMS) and email id. QR code will be shown to the ticket collector where all the booking details can be viewed by scanning the QR code.

4.2. NON-FUNCTIONAL REQUIREMENTS

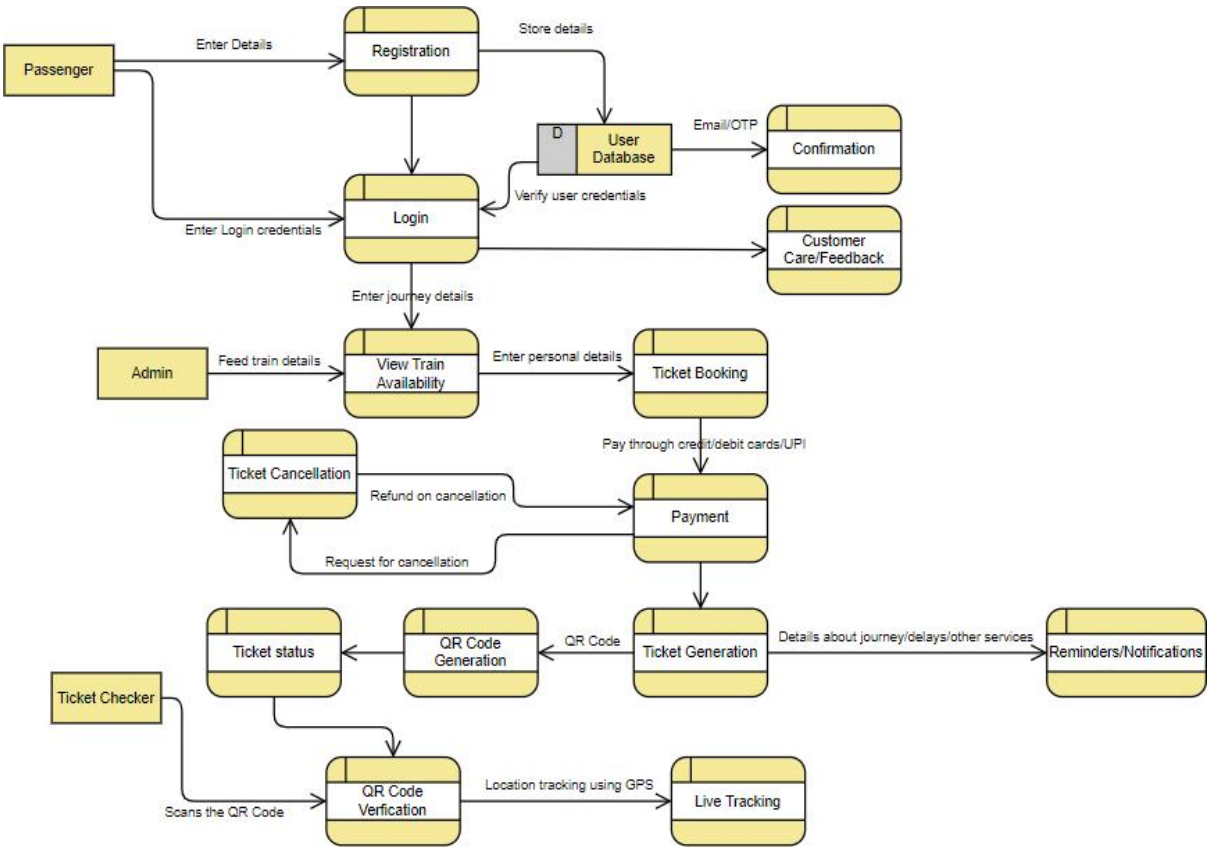
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	This website is a set of easy methods so there will be no difficulties of operating it. All the languages can be accessed through by user.
NFR-2	**Security**	The permission required is only the access of location and there won't be any other unauthorized permission needed.
NFR-3	**Reliability**	If the network connection is disabled While entering the details of user no worries you no need to enter the details again as all the details will be stored automatically.
NFR-4	**Performance**	The website is more secured and it will obtain through the back end. unauthorized person can't access the website.
NFR-5	**Availability**	only the QR code is sended through the message and email id only no other information is included

NFR-6	**Scalability**	At a time more than 300,000 users can obtain .All the data will be stored carefully without any issues.

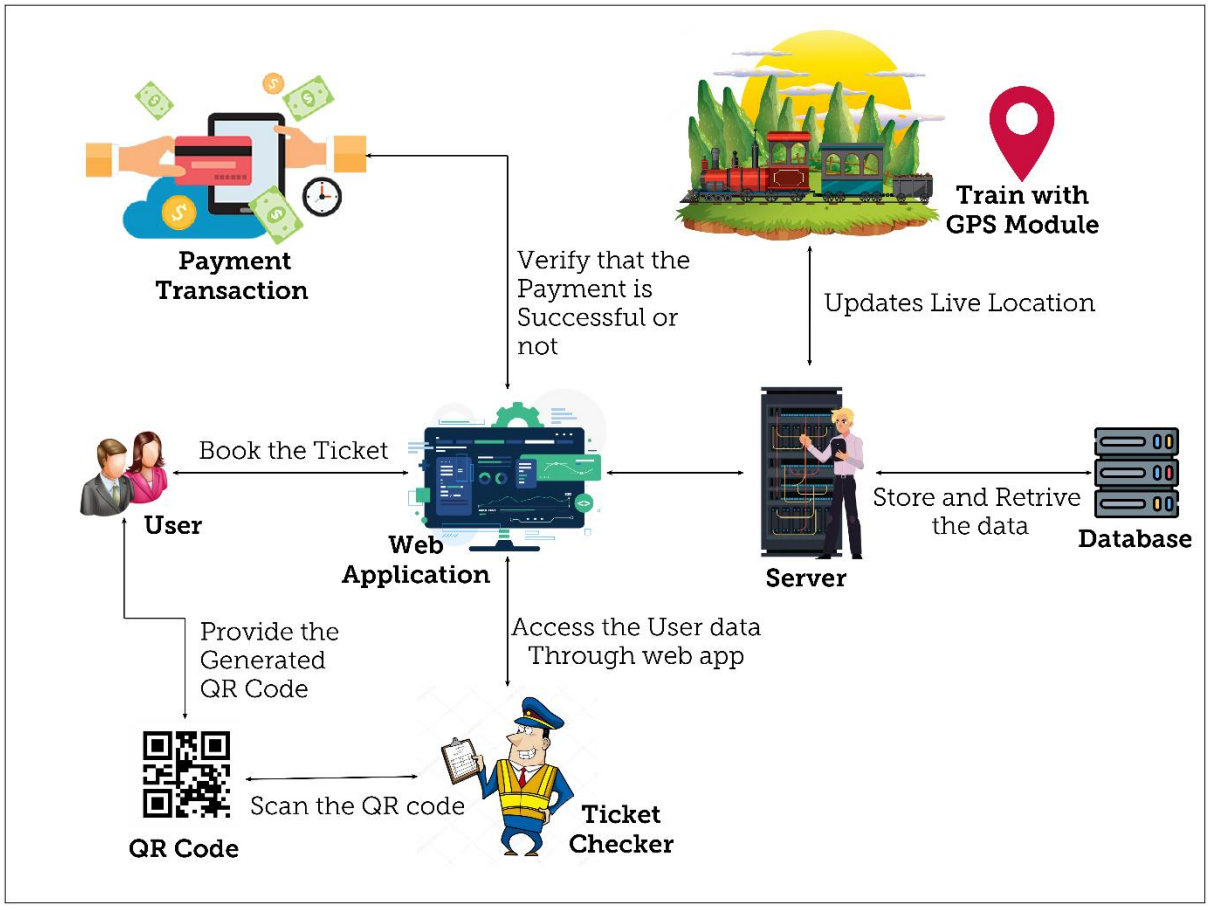
PROJECT DESIGN

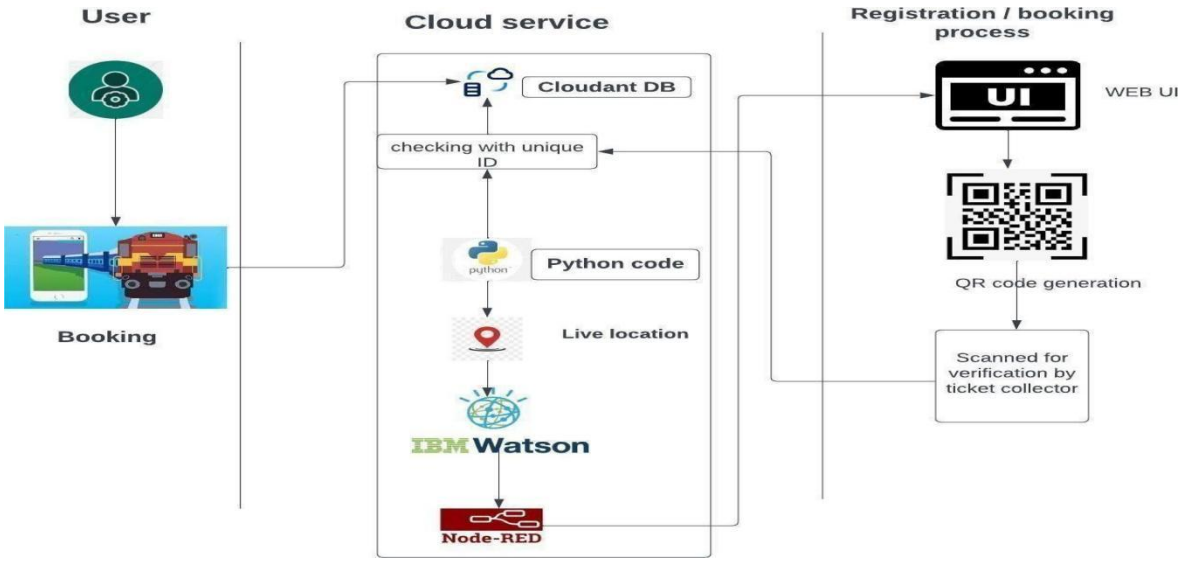
5.PROJECT DESIGN

5.1. DATA FLOW DIAGRAMS



5.2. SOLUTION & TECHNICAL ARCHITECTURE





5.3. USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user, Web user)	Registration	USN-1	As a user, I can register through the form by filling in my details.	I can register and create my account /dashboard	High	Sprint-1
		USN-2	As a user, I can register through phone numbers, gmail, facebook or other social sites.	I can register & create my dashboard with Facebook Login or other social sites	High	Sprint-2
	Confirmation	USN-3	As a user, I will receive confirmation through email or OTP once registration is successful.	I can receive confirmation email & click confirm	High	Sprint-1
	Authentication /Login	USN-4	As a user, I can login via login ID and password or through OTP received on registered phone number.	I can login and access my account/dashboard	High	Sprint-1
	Display train details	USN-5	As a user, I can enter the start and destination to get the list of trains available connecting the above.	I can view the train details(name & number), corresponding routes it passes through based on the start and destination entered.	High	Sprint-1
	Booking	USN-6	As a user, I can provide the basic details such as name, age, gender etc.	I can view,modify or confirm the details entered.	High	Sprint-1
		USN-7	As a user, I can choose the class, seat/berth. If a preferred seat/berth isn't available I can be allotted based on the availability.	I can view, modify or confirm the seats/class/berth selected	High	Sprint-1
	Payment	USN-8	As a user, I can choose to pay through credit card/debit card/UPI.	I can view the payment options available and select my desirable choice to proceed with the payment.	High	Sprint-1
		USN-9	As a user, I will be redirected to the selected payment gateway and upon successful completion of payment	I can pay through the payment portal and confirm the booking.If any	High	Sprint-1

			I'll be redirected to the booking website.	changes need to be done I can move back to the initial payment page.		

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
	Ticket generation	USN-10	As a user, I can download the generated e-ticket for my journey along with the QR code which is used for authentication during my journey.	I can show the generated QR code so that authentication can be done quickly.	High	Sprint-1
	Ticket status	USN-11	As a user, I can see the status of my ticket whether it's confirmed/waiting/RAC.	I can constantly get the information and arrange alternate transport if the ticket isn't confirmed.	High	Sprint-1
	Reminders/Notification	USN-12	As a user, I get reminders about my journey a day before my actual journey.	I can make sure that I don't miss the journey because of the constant notifications.	Medium	Sprint-2
		USN-13	As a user, I can track the train using GPS and can get information such as ETA, current stop and delays.	I can track the train and get to know about the delays and plan accordingly.	Medium	Sprint-2
	Ticket cancellation	USN-14	As a user, I can cancel my ticket if there's any change of plan.	I can cancel the ticket and get a refund based on how close the date is to the commencement of the journey.	High	Sprint-1
	Raise queries	USN-15	As a user, I can raise queries through the query box or via mail.	I can view my previous queries.	Low	Sprint-2
Customer Care Executive	Answer the queries	USN-16	As a user, I will answer the queries/doubts raised by the customers.	I can view the queries and mark it once resolved.	Medium	Sprint-2
Administrator	Feed details	USN-17	As a user, I will feed information about the trains, delays and add extra seats if a new compartment is added.	I can view and ensure the correctness of the information fed.	High	Sprint-1

PROJECT PLANNING AND SCHEDULING

6.PROJECT PLANNING AND SCHEDULING

6.1. SPRINT PLANNING & ESTIMATION

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a Passenger, I can register for the application by entering my email, password, and confirming my password.	7	High	KISHORE KUMAR R
Sprint-1		USN-2	As a Passenger, I will receive confirmation email once I have registered for the application	7	High	KARTHICK ARAVIND E T
Sprint-1	Login	USN-3	As a Passenger, I can log into the application by entering email & password	6	High	KISHORE KUMAR R , DINESH S
Sprint-2	Books Ticket	USN-4	I can select the Train and the train route to be travelled.	4	Medium	KARTHICK ARAVIND E T , DINESH S
Sprint-2		USN-5	I provide the basic details such as name, age, mobile number, etc.	6	High	KISHORE KUMAR R , DINESH KUMAR M
Sprint-2	Selecting the Seat	USN-6	After providing the basic information, I can select the desired seat I wanted if it is in available state.	4	Medium	DINESH KUMAR M , DINESH S
Sprint-2	QR Code Generation	USN-7	At last the QR Code is generated which contains the unique id through which the passenger information can be retrieved.	6	High	KARTHICK ARAVIND E T , DINESH KUMAR M
Sprint-4	Tracking the location of Train	USN-8	As a Passenger, I can track the exact current location of the train.	13	Medium	KISHORE KUMAR R , KARTHICK ARAVIND E T , DINESH S , DINESH KUMAR M
Sprint-3	Login	USN-9	As a Administrator, I can log into the application by entering email & password	6	Medium	KARTHICK ARAVIND E T , DINESH S , DINESH KUMAR M
Sprint-4	Cancel the Booking	USN-10	As a Administrator, I can Cancel the Ticket if the information of the passenger is inappropriate.	7	Low	KISHORE KUMAR R, KARTHICK ARAVIND E T , DINESH S ,

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
						DINESH KUMAR M
Sprint-3	TTR Verifies the Passenger	USN-11	As a Ticket Checker, I can scan the QR Code shown by the passenger.	7	High	KISHORE KUMAR R , KARTHICK ARAVIND E T , DINESH KUMAR M
Sprint-3		USN-12	As a Ticket Checker, I can verify the passenger using the information that displayed after scanning the QR Code.	7	High	KISHORE KUMAR R , KARTHICK ARAVIND E T , DINESH S, DINESH KUMAR M

6.2. SPRINT DELIVERY SCHEDULE

Sprint	*Total Story Points*	*Duration*	*Sprint Start Date*	*Sprint End Date (Planned)*	*Story Points Completed (as on Planned End Date)*	*Sprint Release Date (Actual)*
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	30Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	06 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	13 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	20 Nov 2022

CODING AND SOLUTIONING

7. CODING AND SOLUTIONING

7.1 FEATURE 1

- IOT device
- IBM Watson platform
- Node red
- Cloudant DB
- Web UI
- Geofence
- MIT App
- Python code

7.2. FEATURE 2

- Registration
- Login
- Verification
- Ticket Booking
- Payment
- Ticket Cancellation
- Adding Queries

PROGRAM:

```
labl_0 = Label(base, text="Registration
form",width=20,font=("bold", 20))
labl_0.place(x=90,y=53)
lb1= Label(base, text="Enter Name", width=10, font=("arial",12))
lb1.place(x=20, y=120)
en1= Entry(base)
en1.place(x=200, y=120)
lb3= Label(base, text="Enter Email", width=10, font=("arial",12))
lb3.place(x=19, y=160)
en3= Entry(base)
en3.place(x=200, y=160)
lb4= Label(base, text="Contact Number",
width=13,font=("arial",12))
lb4.place(x=19, y=200)
en4= Entry(base)
en4.place(x=200, y=200)
lb5= Label(base, text="Select Gender", width=15,
font=("arial",12))
lb5.place(x=5, y=240)
var = IntVar()
Radiobutton(base, text="Male", padx=5,variable=var,
value=1).place(x=180, y=240)
Radiobutton(base, text="Female", padx =10,variable=var,
value=2).place(x=240,y=240) 30
Radiobutton(base, text="others", padx=15, variable=var,
```

```
value=3).place(x=310,y=240)
list_of_cntry = ("United States", "India", "Nepal", "Germany")
cv = StringVar()
drplist= OptionMenu(base, cv, *list_of_cntry)
drplist.config(width=15)
cv.set("United States")
lb2= Label(base, text="Select Country",
width=13,font=("arial",12))
lb2.place(x=14,y=280)
drplist.place(x=200, y=275)
lb6= Label(base, text="Enter Password",
width=13,font=("arial",12))
lb6.place(x=19, y=320)
en6= Entry(base, show='*')
en6.place(x=200, y=320)
lb7= Label(base, text="Re-Enter Password",
width=15,font=("arial",12))
lb7.place(x=21, y=360)
en7 =Entry(base, show='*')
en7.place(x=200, y=360)
Button(base, text="Register", width=10).place(x=200,y=400)
base.mainloop() 31
def generateOTP() :
# Declare a digits variable
# which stores all digits
digits = "0123456789"
```

```
OTP = ""
# length of password can be changed
# by changing value in range
for i in range(4) :
OTP += digits[math.floor(random.random() * 10)]
return OTP
# Driver code
if__name__== "__main__" :
print("OTP of 4 digits:", generateOTP())
digits="0123456789"
OTP=""
for i in range(6):
OTP+=digits[math.floor(random.random()*10)]
otp = OTP + " is your OTP"
msg= otp
s = smtplib.SMTP('smtp.gmail.com', 587)
s.starttls()
s.login("Your Gmail Account", "You app password")
emailid = input("Enter your email: ")32
s.sendmail('&&&&&&&&&&&&',emailid,msg)
a = input("Enter Your OTP >>: ")
if a == OTP:
print("Verified")
else:
print("Please Check your OTP again")
```

TESTING

8.1. TEST CASES

SPRINT-1:

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
1	Functional	Registration	registration through the form by Filling in my details		1.Click on register 2.Fill the registration form 3.click Register		Registration form to be filled is to be displayed	Working as expected	Pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
2	UI	Generating OTP	Generating the otp for further process		1.Generating of OTP number		user can register through phone numbers, Gmail, Facebook or other social sites	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
3	Functional	OTP verification	Verify user otp using mail		1.Enter gmail id and enter password 2.click submit	Username: abc@gmail.com password: Testing123	OTP verified is to be displayed	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
4	Functional	Login page	Verify user is able to log into application with InValid credentials		1.Enter into log in page 2.Click on My Account dropdown button 3.Enter InValid username/email in Email text box 4.Enter valid password in password text box 5.Click on login button	Username: abc@gmail password: Testing123	Application should show "Incorrect email or password" validation message.	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
5	Functional	Display Train details	The user can view about the available train details		1.As a user, I can enter the start and destination to get the list of trains available connecting the above	Username: abc@gmail.com password: Testing123678686786876	A user can view about the available trains to enter start and destination details	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R

SPRINT-2:

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
1	Functional	Booking	user can provide the basic details such as a name, age, gender etc		1.Enter method of reservation 2.Enter name,age,gender 3.Enter how many tickets wants to be booked 4.Also enter the number member's details like name,age,gender		Tickets booked to be displayed	Working as expected	Pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
2	UI	Booking seats	User can choose the class, seatberth. If a preferred seatberth isn't available I can be allocated based on the availability		1.known to which the seats are available		known to which the seats are available	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
3	Functional	Payment	user, I can choose to pay through credit Card/debit card/UPI.		1.user can choose payment method 2.pay using tht method		payment for the booked tickets to be done using payment method through either the following methods credit Card/debit card/UPI.	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
4	Functional	Redirection	user can be redirected to the selected		1.After payment the usre will be redirected to the previous page		After payment the usre will be redirected to the previous page	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R

SPRINT-3:

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
1	Functional	Ticket generation	a user can download the generated e ticket for my journey along with the QR code which is used for authentication during my journey.		1.Enter method of reservation 2.Enter name,age,gender 3.Enter how many tickets wants to be booked 4.Also enter the number member's details like name,age,gender		Tickets booked to be displayed	Working as expected	Pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
2	UI	Ticket status	a usercan see the status of my ticket Whether it's confirmed/waiting/RAC		1.known to the status of the tivkets booked		known to the status of the tivkets booked	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
3	Functional	Remainder notification	a user, I get remainders about my journey A day before my actual journey		1.user can get reminder nofication		user can get reminder nofication	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
4	Functional	GPS tracking	user can track the train using GPS and can get information such as ETA, Current stop and delay		1.tracking train for getting information		tracking process through GPS	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R

SPRINT-4:

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	BUG ID	Executed By
1	Functional	Ticket cancellation	user can cancel my tickets there's any Change of plan		1.tickets to be cancelled		Tickets booked to be cancelled	Working as expected	Pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
2	UI	Raise queries	user can raise queries through the query box or via mail.		1.raise the queries		raise the queries	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
3	Functional	Answer the queries	user will answer the questions/doubts Raised by the customers.		1.answer the queries		answer the queries	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R
4	Functional	Feed details	a user will feed information about the trains delays and add extra seats if a new compartment is added.		1.information feeding on trains		information feeding on trains	Working as expected	pass				Dinesh S,Dinesh Kumar M,Karthick Aravind E T,Kishore Kumar R

RESULTS

9.1. PERFORMANCE METRICS

ADVANTAGES &DISADVANTAGES

10.1 ADVANTAGES

- Openness – compatibility between different system modules, potentially from different vendors;
- Orchestration – ability to manage large numbers of devices, with full visibility over them;
- Dynamic scaling – ability to scale the system according to the application needs, through resource virtualization and cloud operation;
- Automation – ability to automate parts of the system monitoring application, leading to better performance and lower operation costs.

10.2.DISADVANTAGES

- Approaches to flexible, effective, efficient, and low-cost data collection for both railway vehicles and infrastructure monitoring, using regular trains;
- Data processing, reduction, and analysis in local controllers, and subsequent sending of that data to the cloud, for further processing;
- Online data processing systems, for real-time monitoring, using emerging
- communication technologies;
- Integrated, interoperable, and scalable solutions for railway systems preventive maintenance.

CONCLUSION AND FUTURE SCOPE

11. CONCLUSION

Accidents occurring in Railway transportation system cost a large number of lives. So this system helps us to prevent accidents and giving information about faults or cracks in advance to railway authorities. So that they can fix them and accidents cases becomes less. This project is cost effective. By using more techniques they can be modified and developed according to their applications. By this system many lives can be saved by avoiding accidents. The idea can be implemented in large scale in the long run to facilitate better safety standards for rail tracks and provide effective testing infrastructure for achieving better results in the future.

12. FUTURE SCOPE

In future CCTV systems with IP based camera can be used for monitoring the visual videos captured from the track. It will also increase security for both passengers and railways. GPS can also be used to detect exact location of track fault area, IP cameras can also be used to show fault with the help of video. Locations on Google maps with the help of sensors can be used to detect in which area track is broken

APPENDIX

13.APPENDIX
13.1. SOURE PROGRAM

```
import math, random
import os
import smtplib
import sqlite3
import requests
from bs4 import BeautifulSoup
from django.contrib.auth.base_user import AbstractBaseUser
from django.db import models
import logging
import pandas as pd
import pyttsx3
from plyer import notification
import time
import numpy as np
import matplotlib.pyplot as plt
from PIL import Image, ImageDraw
from pickle import load,dump
import smtplib, ssl
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
import email
from email import encoders
from email.mime.base import MIMEBase
import attr
```

```
from flask import Blueprint, flash, redirect, request, url_for
from flask.views import MethodView
from flask_babelplus import gettext as _
from flask_login import current_user, login_required
from pluggy import HookimplMarker
from tkinter import*
base = Tk()
base.geometry("500x500")
base.title("registration form")
labl_0 = Label(base, text="Registration
form",width=20,font=("bold",20))

labl_0.place(x=90,y=53)
lb1= Label(base, text="Enter Name", width=10, font=("arial",12))
lb1.place(x=20, y=120)
en1= Entry(base)
en1.place(x=200, y=120)
lb3= Label(base, text="Enter Email", width=10, font=("arial",12))
lb3.place(x=19, y=160)
en3= Entry(base)
en3.place(x=200, y=160)
lb4= Label(base, text="Contact Number",
width=13,font=("arial",12))
lb4.place(x=19, y=200)
en4= Entry(base)
en4.place(x=200, y=200)
```

```
lb5= Label(base, text="Select Gender", width=15,
font=("arial",12))
lb5.place(x=5, y=240)
var = IntVar()
Radiobutton(base, text="Male",
padx=5,variable=var,value=1).place(x=180, y=240)
Radiobutton(base, text="Female", padx
10 ,variable=var,value=2).place(x=240,y=240)
Radiobutton(base, text="others", padx=15, variable=var,
value=3).place(x=310,y=240)
list_of_cntry = ("United States", "India", "Nepal", "Germany")
cv = StringVar()
drplist= OptionMenu(base, cv, *list_of_cntry)
drplist.config(width=15)
cv.set("United States")
lb2= Label(base, text="Select Country",
width=13,font=("arial",12))
lb2.place(x=14,y=280)
drplist.place(x=200, y=275)
lb6= Label(base, text="Enter Password",
width=13,font=("arial",12))
lb6.place(x=19, y=320)
en6= Entry(base, show='*')
en6.place(x=200, y=320)
lb7= Label(base, text="Re-Enter Password",
width=15,font=("arial",12))
```

```
lb7.place(x=21, y=360)
en7 =Entry(base, show='*')
en7.place(x=200, y=360)
Button(base, text="Register", width=10).place(x=200,y=400)
base.mainloop()
def generateOTP() :
# Declare a digits variable
# which stores all digits
digits = "0123456789"
OTP = ""
# length of password can be changed
# by changing value in range
for i in range(4) :
OTP += digits[math.floor(random.random() * 10)]
return OTP
# Driver code
if__name__== "__main__" :
print("OTP of 4 digits:", generateOTP())
digits="0123456789"
OTP=""
for i in range(6):
OTP+=digits[math.floor(random.random()*10)]
otp = OTP + " is your OTP"
msg= otp
s = smtplib.SMTP('smtp.gmail.com', 587)
s.starttls()
```

```
s.login("Your Gmail Account", "You app password")
emailid = input("Enter your email: ")
s.sendmail('&&&&&&&&&&&',emailid,msg)
a = input("Enter Your OTP >>: ")
if a == OTP:
print("Verified")
else:
print("Please Check your OTP again")
root = Tk()
root.title("Python: Simple Login Application")
width = 400
height = 280
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
x = (screen_width/2) - (width/2)
y = (screen_height/2) - (height/2)
root.geometry("%dx%d+%d+%d" % (width, height, x, y))
root.resizable(0, 0)
USERNAME = StringVar()
PASSWORD = StringVar()
Top = Frame(root, bd=2, relief=RIDGE)
Top.pack(side=TOP, fill=X)
Form = Frame(root, height=200)
Form.pack(side=TOP, pady=20)
lbl_title = Label(Top, text = "Python: Simple Login Application",
font=('arial', 15))
```

```
lbl_title.pack(fill=X)
lbl_username = Label(Form, text = "Username:", font=('arial',
14),bd=15)
lbl_username.grid(row=0, sticky="e")
lbl_password = Label(Form, text = "Password:", font=('arial', 14),
bd=15)
lbl_password.grid(row=1, sticky="e")
lbl_text = Label(Form)
lbl_text.grid(row=2, columnspan=2)
username = Entry(Form, textvariable=USERNAME, font=(14))
username.grid(row=0, column=1)
password = Entry(Form, textvariable=PASSWORD, show="*",
font=(14))
password.grid(row=1, column=1)
def Database():
global conn, cursor
conn = sqlite3.connect("pythontut.db")
cursor = conn.cursor()
cursor.execute("CREATE TABLE IF NOT EXISTS `member`
(mem_id INTEGER NOT NULL PRIMARY KEY
AUTOINCREMENT, username TEXT, password TEXT)")
cursor.execute("SELECT * FROM `member` WHERE `username`
='admin' AND `password` = 'admin'") if cursor.fetchone() is None:
cursor.execute("INSERT INTO `member` (username, password)
```

```
VALUES('admin', 'admin')")
conn.commit()
def Login(event=None):
Database()
if USERNAME.get() == "" or PASSWORD.get() == "":
lbl_text.config(text="Please complete the required field!",
fg="red")
else:
cursor.execute("SELECT * FROM `member` WHERE `username`
= ? AND `password` = ?", (USERNAME.get(),
PASSWORD.get()))
if cursor.fetchone() is not None:
HomeWindow()
USERNAME.set("")
PASSWORD.set("")
lbl_text.config(text="")
else:
lbl_text.config(text="Invalid username or password", fg="red")
USERNAME.set("")
PASSWORD.set("")
cursor.close()
conn.close()
btn_login = Button(Form, text="Login", width=45,
command=Login)
btn_login.grid(pady=25, row=3, columnspan=2)
btn_login.bind('<Return>', Login)
```

```
def HomeWindow():
global Home
root.withdraw()
Home = Toplevel()
Home.title("Python: Simple Login Application")
width = 600
height = 500
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
x = (screen_width/2) - (width/2)
y = (screen_height/2) - (height/2)
root.resizable(0, 0)
Home.geometry("%dx%d+%d+%d" % (width, height, x, y))
lbl_home = Label(Home, text="Successfully Login!", font=('times
new
roman', 20)).pack()
btn_back = Button(Home, text='Back',
command=Back).pack(pady=20, fill=X)
def Back():
Home.destroy()
root.deiconify()
def getdata(url):
r = requests.get(url)
return r.text
# input by geek
from_Station_code = "GAYA"
```

```
from_Station_name = "GAYA"
To_station_code = "PNBE"
To_station_name = "PATNA"
# url
url = "https://www.railyatri.in/booking/trains-between
stations?from_code="+from_Station_code+"&from_name="+fro
m_Stat
ion_name+"+JN+&journey_date=+Wed&src=tbs&to_code=" + \
To_station_code+"&to_name="+To_station_name + \
"+JN+&user_id=-
1603228437&user_token=355740&utm_source=dwebsearch_tbs_
search_
trains"
# pass the url
# into getdata function
htmldata = getdata(url)
soup = BeautifulSoup(htmldata, 'html.parser')
# find the Html tag
# with find()
# and convert into string
data_str = ""
for item in soup.find_all("div", class_="col-xs-12
TrainSearchSection"):
data_str = data_str + item.get_text()
result = data_str.split("\n")
```

```
print("Train between "+from_Station_name+" and
"+To_station_name)
print("")
# Display the result
for item in result:
if item != "":
print(item)
print("\n\nTicket Booking System\n")
restart = ('Y')
while restart != ('N','NO','n','no'):
print("1.Check PNR status")
print("2.Ticket Reservation")
option = int(input("\nEnter your option : "))
if option == 1:
print("Your PNR status is t3")
exit(0)
elif option == 2:
people = int(input("\nEnter no. of Ticket you
  want :"))
name_l = []
age_l = []
sex_l = []
for p in range(people):
name = str(input("\nName : "))
name_l.append(name)
age = int(input("\nAge : "))
```

```
age_l.append(age)
sex = str(input("\nMale or Female : "))
sex_l.append(sex)
restart = str(input("\nDid you forgot someone? y/n:
"))
if restart in ('y','YES','yes','Yes'):
restart = ('Y')
else :
x = 0
print("\nTotal Ticket : ",people)
for p in range(1,people+1):
print("Ticket : ",p)
print("Name : ", name_l[x])
print("Age : ", age_l[x])
print("Sex : ",sex_l[x])
x += 1
```

7.2. FEATURE 2

```
'
class User(AbstractBaseUser):
"""
User model.
"""
USERNAME_FIELD = "email"
REQUIRED_FIELDS = ["first_name", "last_name"]
email = models.EmailField(
verbose_name="E-mail",
unique=True
)
first_name =
models.CharField( verbose_na
me="First name",
max_length=30
)
last_name =
models.CharField( verbose_na
me="Last name",
max_length=40
)
city =
models.CharField( verbo
se_name="City",
max_length=40
)
stripe_id =
```

```
models.CharField( verbose_n
ame="Stripe ID",
unique=True, max_length=50,
blank=True, null=True
)
objects = UserManager()
@property
def get_full_name(self):
return f"{self.first_name} {self.last_name}"
class Meta:
verbose_name = "User"
verbose_name_plural = "Users"
class Profile(models.Model):
"""
User's profile.
"""
phone_number =
models.CharField( verbose_name="
Phone number", max_length=15
)
date_of_birth =
models.DateField( verbose_name
="Date of birth"
)
postal_code =
models.CharField( verbose_nam
e="Postal code", max_length=10,
```

```
blank=True
)
address =
models.CharField( verbose_
name="Address",
max_length=255,
blank=True
)
class Meta:
abstract = True
class UserProfile(Profile):
"""
User's profile model.
"""
user = models.OneToOneField(
to=User, on_delete=models.CASCADE, related_name="profile",
)
group =
models.CharField( verbose_name="Group type",
choices=GroupTypeChoices.choices(),
max_length=20,
default=GroupTypeChoices.EMPLOYEE.name,
)
def__str__(self):
return self.user.email
class Meta:
# user 1 - employer
```

```
user1, _ =
User.objects.get_or_create( email="fo
o@bar.com", first_name="Employer",
last_name="Testowy",
city="Białystok",
)
user1.set_unusable_password()
group_name = "employer"
_profile1, _ =
UserProfile.objects.get_or_create( user=user1,
date_of_birth=datetime.now() - timedelta(days=6600),
group=GroupTypeChoices(group_name).name,
address="Myśliwska 14",
postal_code="15-569",
phone_number="+48100200300",
)
# user2 - employee
user2, _ = User.objects.get_or_create()
email="bar@foo.com",
first_name="Employee",
last_name="Testowy",
city="Białystok",
)
user2.set_unusable_password()
group_name = "employee"
_profile2, _ = UserProfile.objects.get_or_create()
```

```
user=user2,
date_of_birth=datetime.now() - timedelta(days=7600),
group=GroupTypeChoices(group_name).name,
address="Myśliwska 14",
postal_code="15-569",
phone_number="+48200300400",
)
response_customer = stripe.Customer.create()
email=user.email,
description=f"EMPLOYER - {user.get_full_name}",
name=user.get_full_name,
phone=user.profile.phone_number,
)
user1.stripe_id = response_customer.stripe_id
user1.save()
mcc_code, url = "1520", "https://www.softserveinc.com/"
response_ca = stripe.Account.create()
type="custom",
country="PL",
email=user2.email,
default_currency="pln",
business_type="individual",
settings={"payouts": {"schedule": {"interval": "manual", }}},
requested_capabilities=["card_payments", "transfers", ],
business_profile={"mcc": mcc_code, "url": url},
individual={
```

```
"first_name": user2.first_name,
"last_name": user2.last_name,
"email": user2.email,
"dob": {
"day": user2.profile.date_of_birth.day,
"month": user2.profile.date_of_birth.month,
"year": user2.profile.date_of_birth.year,
},
"phone": user2.profile.phone_number,
"address": {
"city": user2.city,
"postal_code": user2.profile.postal_code,
"country": "PL",
"line1": user2.profile.address,
},
},
)
user2.stripe_id = response_ca.stripe_id
user2.save()
tos_acceptance = {"date": int(time.time()), "ip": user_ip},
stripe.Account.modify(user2.stripe_id,
tos_acceptance=tos_acceptance)
passport_front =
stripe.File.create( purpose="identi
ty_document", file=_file, #
ContentFile object
stripe_account=user2.stripe_id,
```

```
)
individual =
{ "verification":
{
"document": {"front": passport_front.get("id"),},
"additional_document": {"front": passport_front.get("id"),},
}
}
stripe.Account.modify(user2.stripe_id, individual=individual)
new_card_source = stripe.Customer.create_source(user1.stripe_id,
source=token)
stripe.SetupIntent.create( payment_met
hod_types=["card"],
customer=user1.stripe_id,
description="some description",
payment_method=new_card_source.id,
)
payment_method =
stripe.Customer.retrieve(user1.stripe_id).default_source
payment_intent =
stripe.PaymentIntent.create( amount=amount,
currency="pln", payment_method_types=["card"],
capture_method="manual",
customer=user1.stripe_id, # customer
payment_method=payment_method,
application_fee_amount=application_fee_amount,
```

```
transfer_data={"destination": user2.stripe_id}, # connect account
description=description,
metadata=metadata,
)
payment_intent_confirm =
stripe.PaymentIntent.confirm( payment_intent.stripe_id,
payment_method=payment_method
)
stripe.PaymentIntent.capture( payment_intent.i
d, amount_to_capture=amount
)
stripe.Balance.retrieve(stripe_account=user2.stripe_id)
stripe.Charge.create(
amount=amount,
currency="pln",
source=user2.stripe_id,
description=description
)
stripe.PaymentIntent.cancel(payment_intent.id)
unique_together = ("user", "group") @attr.s(frozen=True,
cmp=False, hash=False, repr=True) class
UserSettings(MethodView):
form = attr.ib(factory=settings_form_factory)
settings_update_handler = attr.ib(factory=settings_update_handler)
decorators = [login_required]
def get(self):
return self.render()
```

```
def post(self):
if self.form.validate_on_submit():
try:
self.settings_update_handler.apply_changeset(
 current_user, self.form.as_change()
)
except StopValidation as e:
self.form.populate_errors(e.reasons)
return self.render()
except PersistenceError:
logger.exception("Error while updating user settings")
flash(_("Error while updating user settings"), "danger")
return self.redirect()
flash(_("Settings updated."), "success")
return self.redirect()
return self.render()
def render(self):
return render_template("user/general_settings.html",
form=self.form)
def redirect(self):
return redirect(url_for("user.settings"))
@attr.s(frozen=True, hash=False, cmp=False, repr=True)
class ChangePassword(MethodView):
form = attr.ib(factory=change_password_form_factory)
password_update_handler =
attr.ib(factory=password_update_handler)
```

```
decorators = [login_required]
def get(self):
return self.render()
def post(self):
if self.form.validate_on_submit():
try:
self.password_update_handler.apply_changeset(
 current_user, self.form.as_change()
)
except StopValidation as e:
self.form.populate_errors(e.reasons)
return self.render()
except PersistenceError:
logger.exception("Error while changing password")
flash(_("Error while changing password"), "danger")
return self.redirect()
flash(_("Password updated."), "success")
return self.redirect()
return self.render()
def render(self):
return render_template("user/change_password.html",
form=self.form)
def redirect(self):
return redirect(url_for("user.change_password"))
@attr.s(frozen=True, cmp=False, hash=False, repr=True)
class ChangeEmail(MethodView):
```

```
form = attr.ib(factory=change_email_form_factory)
update_email_handler = attr.ib(factory=email_update_handler)
decorators = [login_required]
def get(self):
return self.render()
def post(self):
if self.form.validate_on_submit():
try:
self.update_email_handler.apply_changeset(
 current_user, self.form.as_change()
)
except StopValidation as e:
self.form.populate_errors(e.reasons)
return self.render()
except PersistenceError:
logger.exception("Error while updating email")
flash(_("Error while updating email"), "danger")
return self.redirect()
flash(_("Email address updated."), "success")
return self.redirect()
return self.render()
def render(self):
return render_template("user/change_email.html", form=self.form)
def redirect(self):
return redirect(url_for("user.change_email"))
def berth_type(s):
```

```
if s>0 and s<73:
if s % 8 == 1 or s % 8 == 4:
print (s), "is lower berth"
elif s % 8 == 2 or s % 8 == 5:
print (s), "is middle berth"
elif s % 8 == 3 or s % 8 == 6:
print (s), "is upper berth"
elif s % 8 == 7:
print (s), "is side lower berth"
else:
print (s), "is side upper berth"
else:
print (s), "invalid seat number"
# Driver code
s = 10
berth_type(s) # fxn call for berth type
s = 7
berth_type(s) # fxn call for berth type
s = 0
berth_type(s) # fxn call for berth type
class Ticket:
counter=0
def__init__(self,passenger_name,source,destination):
self.__passenger_name=passenger_name
self.__source=source
self.__destination=destination
```

```
self.Counter=Ticket.counter
Ticket.counter+=1
def validate_source_destination(self):
if (self.__source=="Delhi" and (self.__destination=="Pune" or
self.__destination=="Mumbai" or self.__
_____destination=="Chennai" or
self.__destination=="Kolkata")):
return True
else:
return False
def generate_ticket(self ):
if True:
__ticket_id=self.__source[0]+self.__destination[0]+"0"+str(self.C
ounter)
print( "Ticket id will be:",__ticket_id)
else:
return False
def get_ticket_id(self):
return self.ticket_id
def get_passenger_name(self):
return self.__passenger_name
def get_source(self):
if self.__source=="Delhi":
return self.__source
else:
print("you have written invalid soure option")
```

```
return None
def get_destination(self):
if self.__destination=="Pune":
return self.__destination
elif self.__destination=="Mumbai":
return self.__destination
elif self.__destination=="Chennai":
return self.__destination
elif self.__destination=="Kolkata":
return self.__destination
else:
return None
# user define function
# Scrape the data
def getdata(url):
r = requests.get(url)
return r.text
# input by geek
train_name = "03391-rajgir-new-delhi-clone-special-rgd-to-ndls"
# url
url = "https://www.railyatri.in/live-train-status/"+train_name
# pass the url
# into getdata function
htmldata = getdata(url)
soup = BeautifulSoup(htmldata, 'html.parser')
# traverse the live status from
```

```
# this Html code
data = []
for item in soup.find_all('script', type="application/ld+json"):
data.append(item.get_text())
# convert into dataframe
df = pd.read_json(data[2])
# display this column of
# dataframe
print(df["mainEntity"][0]['name'])
print(df["mainEntity"][0]['acceptedAnswer']['text'])
Speak method
def Speak(self, audio):
# Calling the initial constructor
# of pyttsx3
engine = pyttsx3.init('sapi5')
# Calling the getter method
voices = engine.getProperty('voices')
# Calling the setter method
engine.setProperty('voice', voices[1].id)
engine.say(audio) engine.runAndWait()
def Take_break():
Speak("Do you want to start sir?")
question = input()
if "yes" in question:
Speak("Starting Sir")
```

```
if "no" in question:
Speak("We will automatically start after 5 Mins
Sir.")
time.sleep(5*60)
Speak("Starting Sir")
# A notification we will held that
# Let's Start sir and with a message of
# will tell you to take a break after 45
# mins for 10 seconds
while(True):
notification.notify(title="Let's Start sir",
message="will tell you to take a break after 45
mins",
timeout=10)
# For 45 min the will be no notification but
# after 45 min a notification will pop up.
time.sleep(0.5*60)
Speak("Please Take a break Sir")
notification.notify(title="Break Notification",
message="Please do use your device after sometime
as you have"
"been continuously using it for 45 mins and it will
affect your eyes",
timeout=10)
# Driver's Code
if__name__== '__main__':
```

```
Take_break()
data_path = 'data.csv'
data = pd.read_csv(data_path, names=['LATITUDE',
'LONGITUDE'],
sep=',')
gps_data = tuple(zip(data['LATITUDE'].values,
data['LONGITUDE'].values))
image = Image.open('map.png', 'r') # Load map image.
img_points = []
for d in gps_data:
x1, y1 = scale_to_img(d, (image.size[0], image.size[1])) # Convert
GPS
coordinates to image coordinates.
img_points.append((x1, y1))
draw = ImageDraw.Draw(image)
draw.line(img_points, fill=(255, 0, 0), width=2) # Draw converted
records to the map image.
image.save('resultMap.png')
x_ticks = map(lambda x: round(x, 4), np.linspace(lon1, lon2,
num=7))
y_ticks = map(lambda x: round(x, 4), np.linspace(lat1, lat2,
num=8))
y_ticks = sorted(y_ticks, reverse=True) # y ticks must be reversed
due to
conversion to image coordinates.
fig, axis1 = plt.subplots(figsize=(10, 10))
```

```
axis1.imshow(plt.imread('resultMap.png')) # Load the image to
matplotlib plot.
axis1.set_xlabel('Longitude')
axis1.set_ylabel('Latitude')
axis1.set_xticklabels(x_ticks)
axis1.set_yticklabels(y_ticks)
axis1.grid()
plt.show()
class tickets:
def__init__(self):
self.no_ofac1stclass=0
self.totaf=0
self.no_ofac2ndclass=0
self.no_ofac3rdclass=0
self.no_ofsleeper=0
self.no_oftickets=0
self.name=''
self.age=''
self.resno=0
self.status=''
def ret(self):
return(self.resno)
def retname(self):
return(self.name)
def display(self):
f=0
```

```
fin1=open("tickets.dat","rb")
if not fin1:
print "ERROR"
else:
print
n=int(raw_input("ENTER PNR NUMBER : "))
print "\n\n"
print ("FETCHING DATA . . .".center(80))
time.sleep(1)
print
print('PLEASE WAIT...!!'.center(80))
time.sleep(1)
os.system('cls')
try:
while True:
tick=load(fin1)
if(n==tick.ret()):
f=1
print "="*80
print("PNR STATUS".center(80))
print"="*80
print
print "PASSENGER'S NAME :",tick.name
print
print "PASSENGER'S AGE :",tick.age
print
```

```
print "PNR NO :",tick.resno
print
print "STATUS :",tick.status
print
print "NO OF SEATS BOOKED : ",tick.no_oftickets
print
except:
pass
fin1.close()
if(f==0):
print
print "WRONG PNR NUMBER..!!"
print
def pending(self):
self.status="WAITING LIST"
print "PNR NUMBER :",self.resno
print
time.sleep(1.2)
print "STATUS = ",self.status
print
print "NO OF SEATS BOOKED : ",self.no_oftickets
print
def confirmation (self):
self.status="CONFIRMED"
print "PNR NUMBER : ",self.resno
print
```

```
time.sleep(1.5)
print "STATUS = ",self.status
print
def cancellation(self):
z=0
f=0
fin=open("tickets.dat","rb")
fout=open("temp.dat","ab")
print
r= int(raw_input("ENTER PNR NUMBER : "))
try:
while(True):
tick=load(fin)
z=tick.ret()
if(z!=r):
dump(tick,fout)
elif(z==r):
f=1
except:
pass
fin.close()
fout.close()
os.remove("tickets.dat")
os.rename("temp.dat","tickets.dat")
if (f==0):
print
```

```
print "NO SUCH RESERVATION NUMBER FOUND"
print
time.sleep(2)
os.system('cls')
else:
print
print "TICKET CANCELLED"
print"RS.600 REFUNDED...."
def reservation(self):
trainno=int(raw_input("ENTER THE TRAIN NO:"))
z=0
f=0
fin2=open("tr1details.dat")
fin2.seek(0)
if not fin2:
print "ERROR"
else:
try:
while True:
tr=load(fin2)
z=tr.gettrainno()
n=tr.gettrainname()
if (trainno==z):
print
print "TRAIN NAME IS : ",n
f=1
```

```
print
print "-"*80
no_ofac1st=tr.getno_ofac1stclass()
no_ofac2nd=tr.getno_ofac2ndclass()
no_ofac3rd=tr.getno_ofac3rdclass()
no_ofsleeper=tr.getno_ofsleeper()
if(f==1):
fout1=open("tickets.dat","ab")
print
self.name=raw_input("ENTER THE PASSENGER'S
NAME ")
print
self.age=int(raw_input("PASSENGER'S AGE : "))
print
print"\t\t SELECT A CLASS YOU WOULD LIKE TO
TRAVEL IN :- "
print "1.AC FIRST CLASS"
print
print "2.AC SECOND CLASS"
print
print "3.AC THIRD CLASS"
print
print "4.SLEEPER CLASS"78
print
c=int(raw_input("\t\t\tENTER YOUR CHOICE = "))
os.system('cls')
```

```
amt1=0
if(c==1):
self.no_oftickets=int(raw_input("ENTER NO_OF
FIRST CLASS AC SEATS TO BE BOOKED : "))
i=1
while(i<=self.no_oftickets):
self.totaf=self.totaf+1
amt1=1000*self.no_oftickets
i=i+1
print
print "PROCESSING. .",
time.sleep(0.5)
print ".",
time.sleep(0.3)
print'.'
time.sleep(2)
os.system('cls')
print "TOTAL AMOUNT TO BE PAID = ",amt1
self.resno=int(random.randint(1000,2546))
x=no_ofac1st-self.totaf
print
if(x>0):
self.confirmation()
dump(self,fout1)
break
else:
```

```
self.pending()
dump(tick,fout1)
break  elif(c==2):
self.no_oftickets=int(raw_input("ENTER NO_OF
SECOND CLASS AC SEATS TO BE BOOKED : "))
i=1
def menu():
tr=train()
tick=tickets()
print
print "WELCOME TO PRAHIT AGENCY".center(80)
while True:
print
print "="*80
print " \t\t\t\t RAILWAY"
print
print "="*80
print
print "\t\t\t1. **UPDATE TRAIN DETAILS."
print
print "\t\t\t2. TRAIN DETAILS. "
print
print "\t\t\t3. RESERVATION OF TICKETS."
print
print "\t\t\t4. CANCELLATION OF TICKETS. "
```

```
print
print "\t\t\t5. DISPLAY PNR STATUS."
print
print "\t\t\t6. QUIT."
print"** - office use....."
ch=int(raw_input("\t\t\tENTER YOUR CHOICE : "))
os.system('cls')
print
"\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t\t\tLO
ADI
NG. .",
time.sleep(1)
print ("."),
time.sleep(0.5)
print (".")
time.sleep(2)
os.system('cls')
if ch==1:
j="*****"
r=raw_input("\n\n\n\n\n\n\n\n\n\n\n\t\t\t\tENTER THE
PASSWORD: ")
os.system('cls')
if (j==r):
x='y'
while (x.lower()=='y'):
fout=open("tr1details.dat","ab")
```

```
tr.getinput()
dump(tr,fout)
fout.close()
print"\n\n\n\n\n\n\n\n\n\n\n\t\t\tUPDATING TRAIN LIST
PLEASE WAIT . .",
time.sleep(1)
print (".),
time.sleep(0.5)
print (".),
time.sleep(2)
os.system('cls')
print "\n\n\n\n\n\n\n\n\n\n\n"
x=raw_input("\t\tDO YOU WANT TO ADD ANY MORE
TRAINS DETAILS ? ")
os.system('cls')
continue
elif(j<>r):
print"\n\n\n\n\n"
print "WRONG PASSWORD".center(80)
elif ch==2:
fin=open("tr1details.dat",'rb')
if not fin:
print "ERROR"
else:
try:
while True:
```

```
print"*"*80
print"\t\t\t\tTRAIN DETAILS"
print"*"*80
print
tr=load(fin)
tr.output()
raw_input("PRESS ENTER TO VIEW NEXT TRAIN
DETAILS")
os.system('cls')
except EOFError:
pass
elif ch==3:
print'='*80
print "\t\t\t\tRESERVATION OF TICKETS"
print'='*80
print
tick.reservation()
elif ch==4:
print"="*80
print"\t\t\t\tCANCELLATION OF TICKETS"
print
print"="*80
print
tick.cancellation()
elif ch==5:
print "="*80
```

```
print("PNR STATUS".center(80))
print"="*80
printclass tickets:
def__init__(self):
self.no_ofac1stclass=0
self.totaf=0
self.no_ofac2ndclass=0
self.no_ofac3rdclass=0
self.no_ofsleeper=0
self.no_oftickets=0
self.name=" self.age="
self.resno=0
self.status="
def ret(self):
return(self.resno)
def retname(self):
return(self.name)
def display(self):
f=0
fin1=open("tickets.dat","rb")
if not fin1:
print "ERROR"
else:
print
n=int(raw_input("ENTER PNR NUMBER : "))
```

```
print "\n\n"
print ("FETCHING DATA . . .".center(80))
time.sleep(1)
print
print('PLEASE WAIT...!!'.center(80))
time.sleep(1)
os.system('cls')
try:
while True:
tick=load(fin1)
if(n==tick.ret()):
f=1
print "="*80
print("PNR STATUS".center(80))
print"="*80
print84
print "PASSENGER'S NAME :",tick.name
print
print "PASSENGER'S AGE :",tick.age
print
print "PNR NO :",tick.resno
print
print "STATUS :",tick.status
print
print "NO OF SEATS BOOKED : ",tick.no_oftickets
print
```

```
except:
pass
fin1.close()
if(f==0):
print
print "WRONG PNR NUMBER..!!"
print
def pending(self):
self.status="WAITING LIST"
print "PNR NUMBER :",self.resno
print
time.sleep(1.2)
print "STATUS = ",self.status
print
print "NO OF SEATS BOOKED : ",self.no_oftickets
print
def confirmation (self):
self.status="CONFIRMED"
print "PNR NUMBER : ",self.resno
print
time.sleep(1.5)
print "STATUS = ",self.status
print
def cancellation(self):
z=0
f=0
```

```
fin=open("tickets.dat","rb")
fout=open("temp.dat","ab")
print
r= int(raw_input("ENTER PNR NUMBER : "))
try:
while(True):
tick=load(fin)
z=tick.ret()
if(z!=r):
dump(tick,fout)
elif(z==r):
f=1
except:
pass
fin.close()
fout.close()
os.remove("tickets.dat")
os.rename("temp.dat","tickets.dat")
if (f==0):
print
print "NO SUCH RESERVATION NUMBER FOUND"
print
time.sleep(2)
os.system('cls')
else:
print
```

```
print "TICKET CANCELLED"
print"RS.600 REFUNDED...."
def reservation(self):
trainno=int(raw_input("ENTER THE TRAIN NO:"))
z=0
f=0
fin2=open("tr1details.dat")
fin2.seek(0)
if not fin2:
print "ERROR"
else:
try:
while True:
tr=load(fin2)
z=tr.gettrainno()
n=tr.gettrainname()
if (trainno==z):
print
print "TRAIN NAME IS : ",n
f=1
print
print "-"*80
no_ofac1st=tr.getno_ofac1stclass()
no_ofac2nd=tr.getno_ofac2ndclass()
no_ofac3rd=tr.getno_ofac3rdclass()
no_ofsleeper=tr.getno_ofsleeper()
```

```
if(f==1):
fout1=open("tickets.dat","ab")
print
self.name=raw_input("ENTER THE PASSENGER'S
NAME ")
print
self.age=int(raw_input("PASSENGER'S AGE : "))
print
print"\t\t SELECT A CLASS YOU WOULD LIKE TO
TRAVEL IN :- "
print "1.AC FIRST CLASS"
print
print "2.AC SECOND CLASS"
print
print "3.AC THIRD CLASS"
print
print "4.SLEEPER CLASS"
print
c=int(raw_input("\t\t\tENTER YOUR CHOICE = "))
os.system('cls')
amt1=0
if(c==1):
self.no_oftickets=int(raw_input("ENTER NO_OF
FIRST CLASS AC SEATS TO BE BOOKED : "))
i=1
while(i<=self.no_oftickets):
```

```
self.totaf=self.totaf+1
amt1=1000*self.no_oftickets
i=i+1
print
print "PROCESSING. .",
time.sleep(0.5)
print ".",
time.sleep(0.3)
print'.'
time.sleep(2)
os.system('cls')
print "TOTAL AMOUNT TO BE PAID = ",amt1
self.resno=int(random.randint(1000,2546))
x=no_ofac1st-self.totaf
print
if(x>0):
self.confirmation()
dump(self,fout1)
break
else:
self.pending()
dump(tick,fout1)
break  elif(c==2):
self.no_oftickets=int(raw_input("ENTER NO_OF
SECOND CLASS AC SEATS TO BE BOOKED : "))
```

```
i=1
def menu():
tr=train()
tick=tickets()
print
print "WELCOME TO PRAHIT AGENCY".center(80)
while
True:print
print "="*80
print " \t\t\t\t RAILWAY"
print
print "="*80
print
print "\t\t\t1. **UPDATE TRAIN DETAILS."
print
print "\t\t\t2. TRAIN DETAILS. "
print
print "\t\t\t3. RESERVATION OF TICKETS."
print
print "\t\t\t4. CANCELLATION OF TICKETS. "
print
print "\t\t\t5. DISPLAY PNR STATUS."
print
print "\t\t\t6. QUIT."
print"** - office use....."
ch=int(raw_input("\t\t\tENTER YOUR CHOICE : "))
```

```
os.system('cls')
print
"\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t\t\tLO
ADI
NG. .",
time.sleep(1)
print (".",
time.sleep(0.5)
print (".")
time.sleep(2)
os.system('cls')
if ch==1:
j="*****"
r=raw_input("\n\n\n\n\n\n\n\n\n\n\t\t\t\tENTER THE
PASSWORD: ")
os.system('cls')
if (j==r):
x='y'
while (x.lower()=='y'):
fout=open("tr1details.dat","ab")
tr.getinput()
dump(tr,fout)
fout.close()
print"\n\n\n\n\n\n\n\n\n\n\t\t\tUPDATING TRAIN LIST
PLEASE WAIT . .",
time.sleep(1)
```

```
print ("."),
time.sleep(0.5)
print ("."),
time.sleep(2)
os.system('cls')
print "\n\n\n\n\n\n\n\n\n\n\n\n"
x=raw_input("\t\tDO YOU WANT TO ADD ANY MORE
TRAINS DETAILS ? ")
os.system('cls')
continue
elif(j<>r):
print"\n\n\n\n\n"
print "WRONG PASSWORD".center(80)
elif ch==2:
fin=open("tr1details.dat",'rb')
if not fin:
print "ERROR"
tick.display()
elif ch==6:
quit()
raw_input("PRESS ENTER TO GO TO BACK
MENU".center(80))
os.system('cls')
menu()
sender_email = "my@gmail.com"
receiver_email = "your@gmail.com"
```

```
password = input("Type your password and press enter:")
message = MIMEMultipart("alternative")
message["Subject"] = "multipart test"
message["From"] = sender_email
message["To"] = receiver_email
# Create the plain-text and HTML version of your message
text = """\
Hi,
How are you?
Real Python has many great tutorials:
www.realpython.com"""
html = """\
<html>
<body>
<p>Hi,<br>
How are you?<br>92
<a href="http://www.realpython.com">Real Python</a>
has many great tutorials.
</p>
</body>
</html>
"""
# Turn these into plain/html MIMEText objects
part1 = MIMEText(text, "plain")
part2 = MIMEText(html, "html")
# Add HTML/plain-text parts to MIMEMultipart message
```

```
# The email client will try to render the last part first
message.attach(part1)
message.attach(part2)
# Create secure connection with server and send email
context = ssl.create_default_context()
with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context)
as
server:
server.login(sender_email, password)
server.sendmail(
sender_email, receiver_email, message.as_string()
)
subject = "An email with attachment from Python"
body = "This is an email with attachment sent from Python"
sender_email = "my@gmail.com"
receiver_email = "your@gmail.com"
password = input("Type your password and press enter:") #
Create a multipart message and set headers
message = MIMEMultipart()
message["From"] = sender_email
message["To"] = receiver_email
message["Subject"] = subject
message["Bcc"] = receiver_email # Recommended for mass
emails
# Add body to email
message.attach(MIMEText(body, "plain"))
```

```
filename = "document.pdf" # In same directory as script
# Open PDF file in binary mode
with open(filename, "rb") as attachment:
# Add file as application/octet-stream
# Email client can usually download this automatically as
attachment
part = MIMEBase("application", "octet-stream")
part.set_payload(attachment.read())
# Encode file in ASCII characters to send by email
encoders.encode_base64(part)
# Add header as key/value pair to attachment part
part.add_header(
"Content-Disposition",
f"attachment; filename= {filename}",
)
# Add attachment to message and convert message to string
message.attach(part)
text = message.as_string()
# Log in to server using secure context and send email
context = ssl.create_default_context()
with smtplib.SMTP_SSL("smtp.gmail.com", 465, context=context)
as
server:
server.login(sender_email, password)
server.sendmail(sender_email, receiver_email, text)
api_key = "Your_API_key"
```

```
# base_url variable to store url
base_url = "https://api.railwayapi.com/v2/pnr-status/pnr/"
# Enter valid pnr_number
pnr_number = "6515483790"
# Stores complete url address
complete_url = base_url + pnr_number + "/apikey/" + api_key +
"/"
# get method of requests module
# return response object
response_ob = requests.get(complete_url)
# json method of response object convert
# json format data into python format data
result = response_ob.json()
# now result contains list
# of nested dictionaries
if result["response_code"] == 200: #
train name is extracting
# from the result variable data
train_name = result["train"]["name"]
# train number is extracting from
# the result variable data
train_number = result["train"]["number"]
# from station name is extracting
# from the result variable data
from_station = result["from_station"]["name"]
# to_station name is extracting from
```

```
# the result variable data
to_station = result["to_station"]["name"]
# boarding point station name is
# extracting from the result variable data
boarding_point = result["boarding_point"]["name"]
# reservation upto station name is
# extracting from the result variable data
reservation_upto =
result["reservation_upto"]["name"]
# store the value or data of "pnr"
# key in pnr_num variable
pnr_num = result["pnr"]
# store the value or data of "doj" key
# in variable date_of_journey variable
date_of_journey = result["doj"]
# store the value or data of
# "total_passengers" key in variable
total_passengers = result["total_passengers"]
# store the value or data of "passengers"
# key in variable passengers_list
passengers_list = result["passengers"]
# store the value or data of
# "chart_prepared" key in variable
chart_prepared = result["chart_prepared"]
# print following values
print(" train name : " + str(train_name)
```

```
+ "\n train number : " + str(train_number)
+ "\n from station : " + str(from_station)
+ "\n to station : " + str(to_station)
+ "\n boarding point : " + str(boarding_point)
+ "\n reservation upto : " + str(reservation_upto)
+ "\n pnr number : " + str(pnr_num)
+ "\n date of journey : " + str(date_of_journey)
+ "\n total no. of passengers: " +
str(total_passengers)
+ "\n chart prepared : " + str(chart_prepared))
# looping through passenger list
for passenger in passengers_list:
# store the value or data
# of "no" key in variable
passenger_num = passenger["no"]
# store the value or data of
# "current_status" key in variable
current_status = passenger["current_status"]
# store the value or data of
# "booking_status" key in variable
booking_status = passenger["booking_status"]
# print following values
print(" passenger number : " + str(passenger_num)
+ "\n current status : " + str(current_status)
+ "\n booking_status : " + str(booking_status))
else: print("Record Not Found")
```

13.2. GITHUB LINK:

https://github.com/IBM-EPBL/IBM-Project-3406-1658559748