

SMART RAILWAYS

The new era of train travel

PROJECT BASED EXPERIENTIAL LEARNING PROGRAM (NALAIYA THIRAN)

SMART RAILWAYS

A PROJECT REPORT

Submitted by

Kabilan S	(111719106064)
Kirankumar D	(111719106072)
K. Sai krishna	(111719106075)
Naresh S	(111719106100)

TEAM ID : PNT2022TMID16036

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION
ENGINEERING**

R.M.K.ENGINEERING COLLEGE

(An Autonomous Institution)

R.S.M. Nagar, Kavaraipettai-601 206



NOVEMBER 2022

Problem Statement

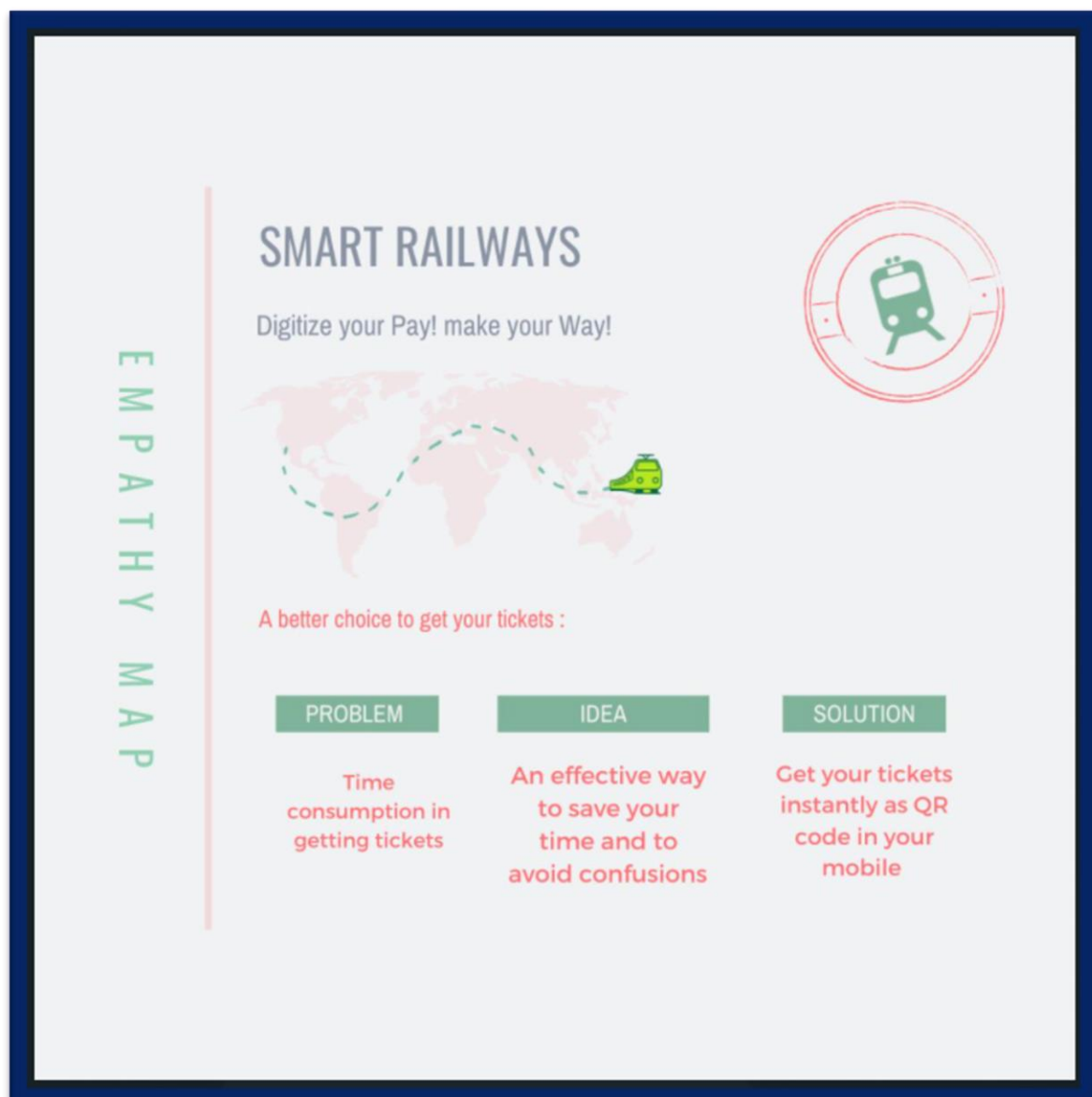
To provide a smart way for booking tickets in railway department through a webpage with a unique QR for each ticket and to deliver the live status of the train to the passengers which is helpful in the critical situations (Stuck of train in forest areas)

Abstract & Ideation

A queue occurs when there are more customers than employees to serve them. This means that customers have to wait for their turn. Whether the waiting itself is an issue or not can only be determined by the customers. A crucial factor in solving a queuing problem is managing the customer's perceived service level.

Solving queuing problems should be a top priority of any service provider. By ensuring that the right customer is at the right place, at the right time, and served by the most appropriate staff, organizations can;

- Increase sales and productivity by up to 30%
- Decrease costs by up to 30%.



Novelty

- Efficient booking system by verifying and validating the ticket as only registered users can book the tickets.
- Each passengers will be provided by a unique ID to them during first login so that their data will be stored and processed securely.
- GPS tracking facility is provided to track the current location of the train from any place.
- A chat box will be provided for the passengers to post their queries or their needs and that will be fulfilled as soon as possible

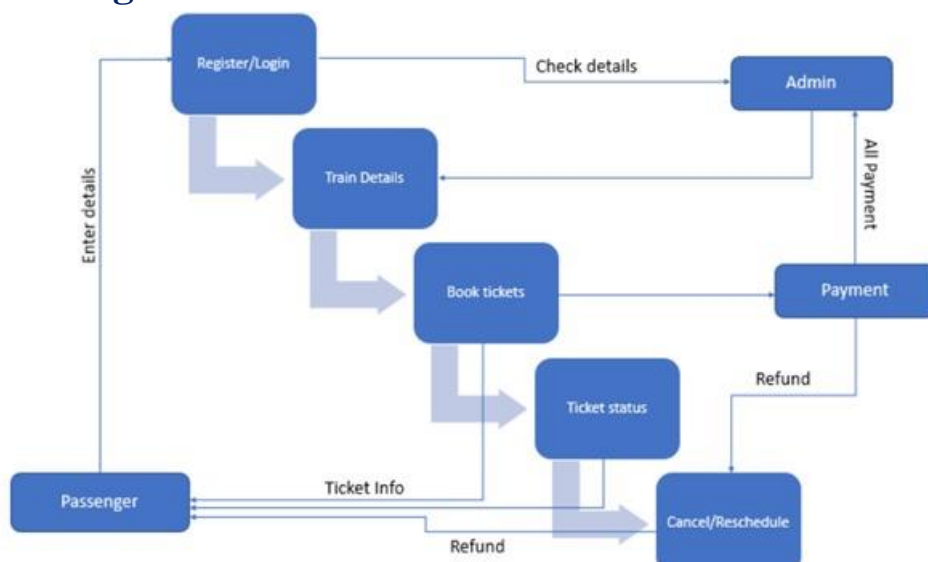
Literature survey

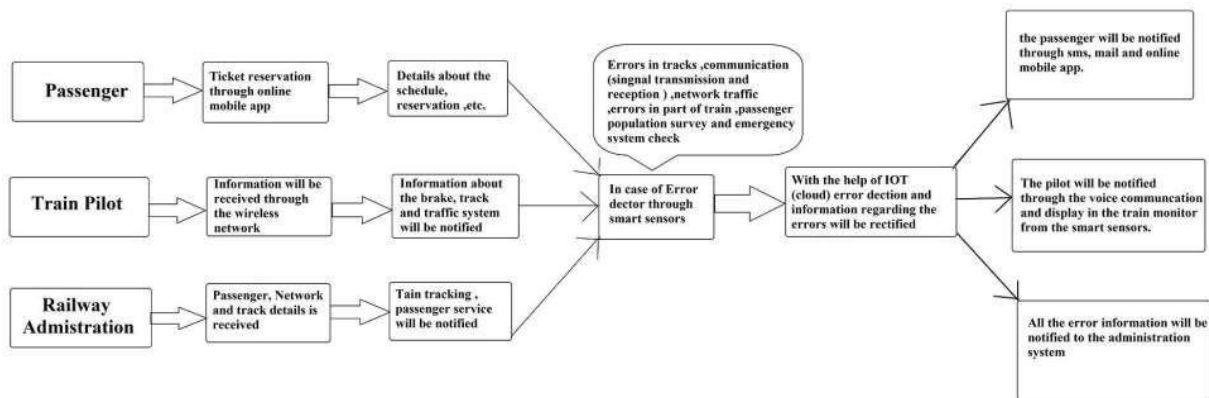
Railways is the major transport among the various modes of transports in the world. It is cost efficient and most preferred by people. Recently Indian railways have started using the latest technology for making the service more efficient by providing various services to its passengers to make their journey comfortable. But still there are a few drawbacks causing much trouble to the passengers.

References:

1. Roman Khoebal, Teeravisit Laohapensaeng, Rounsang Chaisrichaen, "Passenger Monitoring Model for easily Accessible Public City Trams/Trains" (2015).
2. Parag Chatterjee, Asoke Nath, "Application of smart computing in Indian Railway Systems" (2014).
3. Sana Khoja, Maithili Kadam, "Android Suburban Railway Ticketing with GPS as Ticket Checker" (2012).
4. Sujith Kumar, K.M.Yatheendra Parvan, V.Sumathy, Thejeswari C.K, "Novel Approach for Smart Indian Railways" (2017).
5. Sarvath Saba, Sharon Philip, Shriharsha, Mukund Naik, Sudeep Sherry, "A Review on IOT based automated seat allocation and verification using QR code"(2022).

Data flow diagram





Requirements

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Before the user registration there will be options to select the language. All the language is applicable, when user enter in to the application they can see the page of showing enter the email, mobile number and name. After that in screen it shows the verification code is sent through the email id.
FR-2	User verification	The verification code is sent to the registered mail ID
FR-3	User confirmation	The verification code is entered in the application. After finishing, home page opens up.
FR-4	Process of booking	When the home page is opened there will be a from and to option. 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 google pay, phone pay, Paytm, etc. When we select that method, it processes through selected payment option then payment should be done carefully, then the ticket is confirmed. After confirmation it will return to the page and we can see the details of booking.
FR-6	Confirmation message	After all the QR code will be send through the SMS and email id. QR code will be shown to the ticket collector when the QR code is scanned booking details will be shown.

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The app can be accessed very easily by the user. Even they can choose their own languages of comfort.
NFR-2	Security	The permissions access is only for the location access only there will be no other unauthorized permission should be entered to it.
NFR-3	Reliability	While entering the details , if there is any interruption like network disabled etc. the process will be stored automatically and after the connection recovers the user can continue their process
NFR-4	Performance	Application is created in a secure way so that no unauthorized user can access the data from the backend.
NFR-5	Availability	The user will only be provided with the QR code to their registered mail ID
NFR-6	Scalability	At a time more than 300,000 users can use the application. All the data will be stored in the cloud simultaneously.

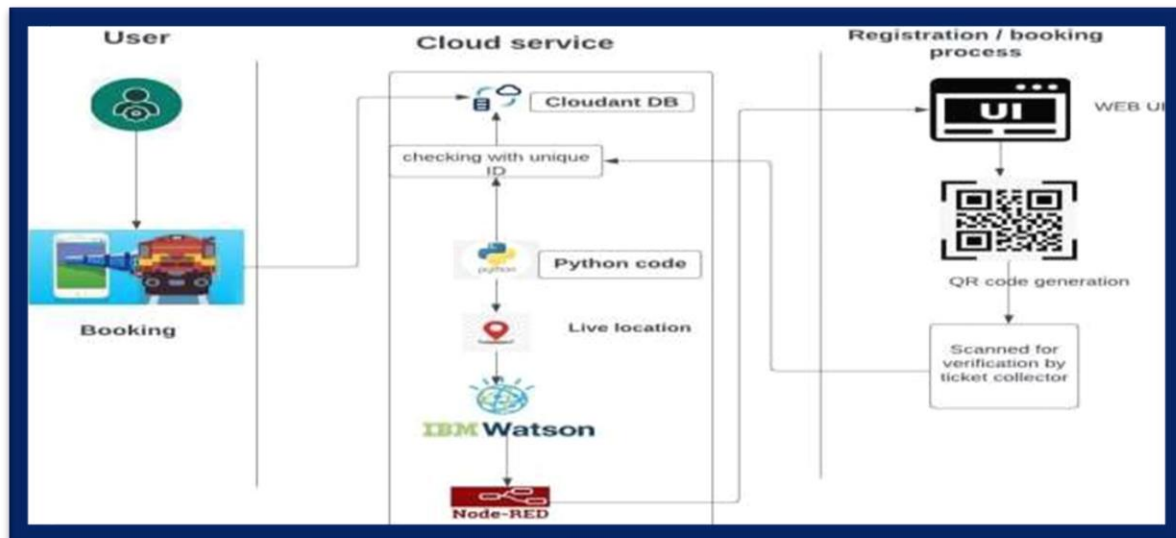
Software requirements

- Python IDE
- IBM Cloud
- IBM Watson IOT Platform
- Node-RED

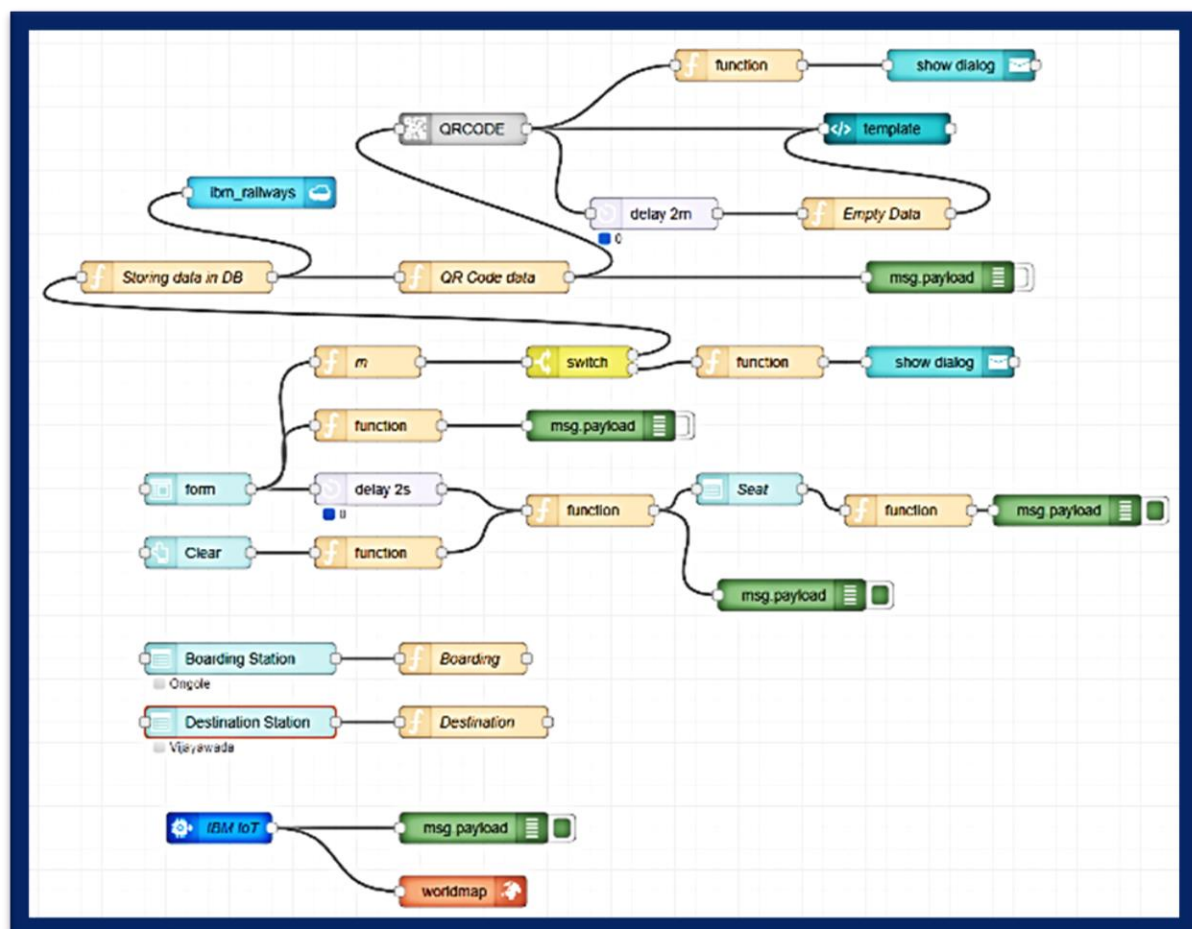
Components & Technologies

- Web UI - HTML, CSS, JavaScript
- Cloud Services - Python
- GPS Tracking - IBM Watson Service
- External API-1 - Sabre API
- External API-2 - Trainline B2B API

Technological Architecture



Node-RED Connection



Source Code

QR SCAN CODE:

```
from http import client
import cv2
import pyzbar
from pyzbar.pyzbar import decode
import time

from ibmcloudant.cloudant_v1 import CloudantV1
from ibmcloudant import CouchDbSessionAuthenticator
from ibm_cloud_sdk_core.authenticators import BasicAuthenticator

authenticator = BasicAuthenticator('apikey-v2-geQfaxlw_D2ghKgN2jjxMsM9giOzQd8yuULxxA4pRH7B',
'daf3cooc2cc182af425a5691a07f7b93')
service = CloudantV1(authenticator=authenticator)

service.set_service_url('https://apikey-v2-geQfaxlw_D2ghKgN2jjxMsM9giOzQd8yuULxxA4pRH7B:daf3cooc2cc182af425a5691a07f7b93@932393aa-9f82-4144-9251-2c519fb30962-bluemix.cloudantnosqldb.appdomain.cloud')

cap= cv2.VideoCapture(0)
font = cv2.FONT_HERSHEY_PLAIN

while True:
    _, frame = cap.read()
    decodedObjects = decode(frame)
    for obj in decodedObjects:
        #print ("Data", obj.data)
        a=obj.data.decode('UTF-8')
        cv2.putText(frame, "Ticket", (50, 50), font, 2, (255, 0, 0), 3)

        #print (a)
        try:
            response = service.get_document(
                db='IBM_railways',
                doc_id = a
            ).get_result()
            print (response)
            time.sleep(5)
        except Exception as e:
            print(a)
            print ("Not a Valid Ticket")
            time.sleep(5)

    cv2.imshow("Frame",frame)
    if cv2.waitKey(1) & 0xFF ==ord('q'):
        break
    cap.release()
cv2.destroyAllWindows()
client.disconnect()
```

LOCATION CODE

```
import wiotp.sdk.device
import time
import random

myConfig = {
    "identity": {
        "orgId": "g2owc1",
        "typeId": "h2",
        "deviceId": "h2"
    },
    "auth": {
```



```

    "token": "12345678"
  }
}

def myCommandCallback(cmd):
    print("The Message received from IBM IoT Platform is : %s" % cmd.data['command'])
    m=cmd.data['command']

def pub(data):
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)
    print("Data is published Successfully:%s",myData)

client = wiotp.sdk.device.DeviceClient(config=myConfig)
client.connect()

while True:
    myData={'name':'Train1','lat':10.184363,'lon': 77.922702}
    pub(myData)
    time.sleep(3)
    myData={'name':'Train1','lat':10.213225,'lon': 77.898765}
    pub(myData)
    time.sleep(3)
    myData={'name':'Train1','lat':10.285035,'lon': 77.921569}
    pub(myData)
    time.sleep(3)
    myData={'name':'Train1','lat':10.343369,'lon': 77.958056}
    pub(myData)
    time.sleep(3)
    myData={'name':'Train1','lat':10.356829,'lon': 77.980861}
    pub(myData)
    time.sleep(3)
    client.commandCallback = myCommandCallback
    client.disconnect()

```

[GITHUB LINK](https://github.com/IBM-EPBL/IBM-Project-31080-1660196047) - <https://github.com/IBM-EPBL/IBM-Project-31080-1660196047>

Testing

Form


Boarding Station

Chennai

Destination

Combatore

QR Code



Clear

CLEAR

Seat

Select option

Details

Name *

Age *

Mobile.No *

CANCEL

Form

Boarding Station

Chennai

Destination

Combatore

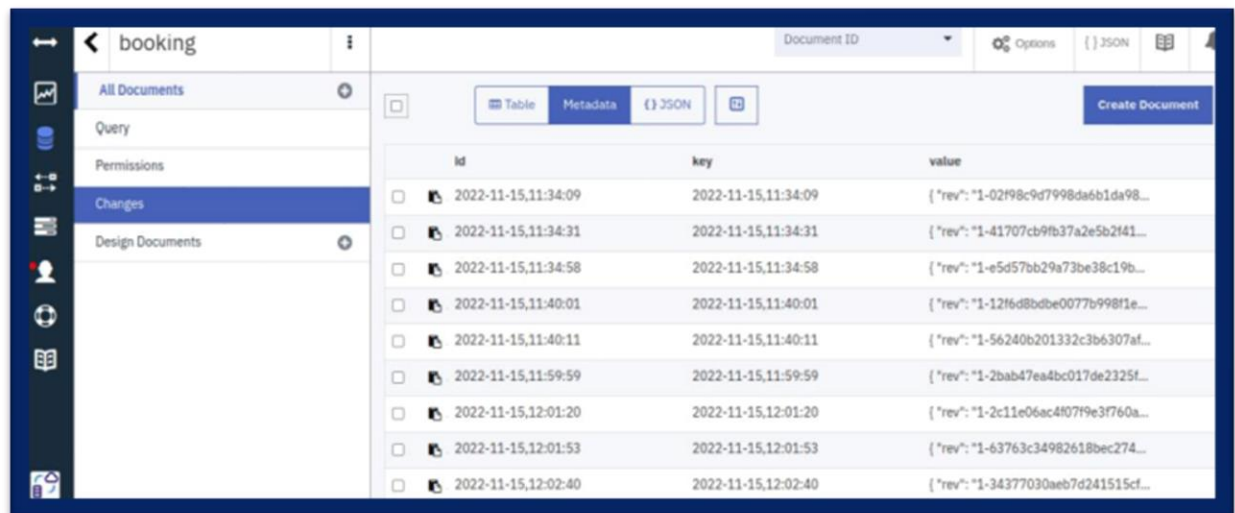
QR Code



Clear

CLEAR

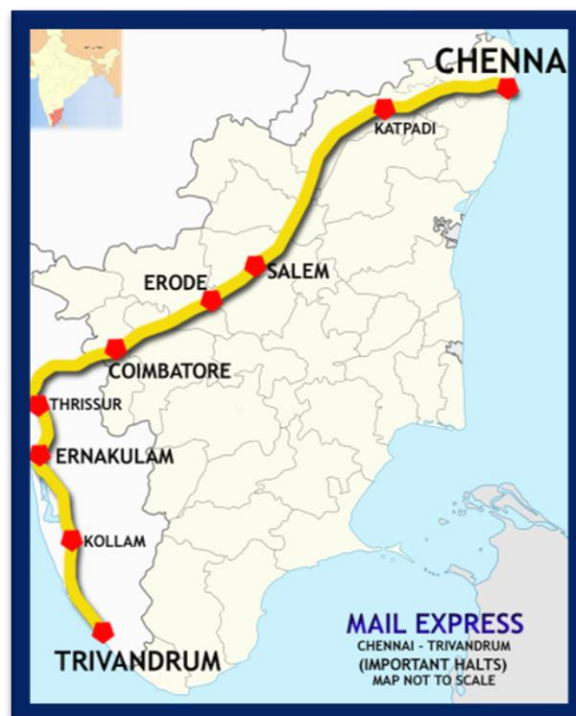
Database Overview



The screenshot shows a database management interface for a 'booking' collection. On the left is a sidebar with navigation options: All Documents, Query, Permissions, Changes (highlighted), and Design Documents. The main area displays a table view of the data. At the top right, there are controls for 'Document ID', 'Options', and a 'Create Document' button. The table has three columns: 'id', 'key', and 'value'. Each row represents a booking record with a timestamp in the 'id' and 'key' columns, and a JSON object in the 'value' column containing a 'rev' field.

id	key	value
<input type="checkbox"/> 2022-11-15,11:34:09	2022-11-15,11:34:09	{ "rev": "1-02f98c9d7998da6b1da98..." }
<input type="checkbox"/> 2022-11-15,11:34:31	2022-11-15,11:34:31	{ "rev": "1-41707cb9fb37a2e5b2f41..." }
<input type="checkbox"/> 2022-11-15,11:34:58	2022-11-15,11:34:58	{ "rev": "1-e5d57bb29a73be38c19b..." }
<input type="checkbox"/> 2022-11-15,11:40:01	2022-11-15,11:40:01	{ "rev": "1-12f6d8bde0077b998f1e..." }
<input type="checkbox"/> 2022-11-15,11:40:11	2022-11-15,11:40:11	{ "rev": "1-56240b201332c3b6307af..." }
<input type="checkbox"/> 2022-11-15,11:59:59	2022-11-15,11:59:59	{ "rev": "1-2bab47ea4bc017de2325f..." }
<input type="checkbox"/> 2022-11-15,12:01:20	2022-11-15,12:01:20	{ "rev": "1-2c11e06ac4f07f9e3f760a..." }
<input type="checkbox"/> 2022-11-15,12:01:53	2022-11-15,12:01:53	{ "rev": "1-63763c34982618bec274..." }
<input type="checkbox"/> 2022-11-15,12:02:40	2022-11-15,12:02:40	{ "rev": "1-34377030aeb7d241515cf..." }

GPS Overview



Conclusion

Using the application, user can book train tickets based on availability of seats in particular train. Once tickets are available, they can book them by inputting their general information. Upon completion of payment, the data gets stored in Cloudant DB with unique ID for every transaction and a QR code is generated for every ticket. The ticket collector can scan the QR code to get information of the passenger, if the QR is correct, the details of the user are displayed, if the QR is invalid, it displays “Not a Valid ticket”. Apart from ticketing, our application also allows the users to find out the live location and running status of the train.

Future Scope

Cloud computing and IOT are integrated now to ease the ticketing system and tracking in railways. In near future, Internet of Things and Artificial Intelligence can be combined to make railways safer and faster. Artificial Intelligence can be used to determine delay and arrival time so that the passenger can act accordingly. By the use of Internet of Things, things such as maintenance of tracks, repairs and services can be carried out with ease.