IOT- PUBLIC TRANSPORTATION AND OPTIMIZATION

PHASE 4

AIM

The aim of public transportation optimization is to improve the efficiency, accessibility, and sustainability of public transportation systems. There are several key objectives and goals associated with public transport optimization

1. **Increased Efficiency:** Public transport optimization aims to make public transportation systems more efficient by reducing travel times, minimizing delays, and increasing the overall speed and reliability of services. This can involve optimizing routes, schedules, and infrastructure.
2. **Reducing Congestion:** One of the primary aims is to alleviate traffic congestion and reduce the number of private vehicles on the road. By providing efficient and attractive public transportation options, more people are encouraged to use public transit instead of driving.
3. **Cost-Effectiveness:** Optimization also involves finding ways to make public transportation more cost-effective. This might involve reducing operational costs, increasing revenue, or finding ways to fund public transportation systems through various means such as subsidies, fares, or public-private partnerships.
4. **Accessibility:** Ensuring that public transportation is accessible to all members of the community is a key goal. This includes making

sure that people with disabilities can easily use the system and that public transit reaches all neighborhoods, not just urban centers.

5. **Environmental Sustainability:** Reducing the environmental impact of public transportation is a significant aim. This can involve transitioning to cleaner and more energy-efficient modes of transport, such as electric buses or trains, and implementing strategies to reduce emissions and the overall carbon footprint.
6. **Safety:** Ensuring the safety of passengers and reducing accidents is an important goal. This includes improving the safety of transportation infrastructure and vehicles.
7. **User Experience:** Public transport optimization aims to provide a positive user experience. This involves making public transportation more convenient, comfortable, and reliable. It may also involve integrating technology for real-time updates, ticketing, and trip planning.
8. **Ridership Growth:** Ultimately, the goal is to increase ridership, getting more people to use public transportation. This not only benefits the environment but also reduces congestion and supports economic development

PROGRAM

IN PYCHARM

```
# This docker-compose file starts and runs:

# * A 3-node kafka cluster

# * A 1-zookeeper ensemble

# * Schema Registry
# * Kafka REST Proxy
# * Kafka Connect
#

version: '3.7'

services:
  zookeeper:
    image: confluentinc/cp-zookeeper:5.2.2
    ports:
    - "2181:2181"
    environment:
      ZOOKEEPER_CLIENT_PORT: "2181"

  kafka0:
    image: confluentinc/cp-kafka:5.2.2
    ports:
    - "9092:9092"
    environment:
      KAFKA_BROKER_ID: 0
      KAFKA_ZOOKEEPER_CONNECT: "zookeeper:2181"
      KAFKA_ADVERTISED_LISTENERS: "INTERNAL://kafka0:19092,EXTERNAL://${DOCKER_HOST_IP:-127.0.0.1}:9092"
      KAFKA_INTER_BROKER_LISTENER_NAME: "INTERNAL"
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP: "INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT"
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: "1"
    depends_on:
    - "zookeeper"

  schema-registry:
    image: confluentinc/cp-schema-registry:5.2.2
    ports:
    - "8081:8081"
    environment:
      SCHEMA_REGISTRY_KAFKASTORE_BOOTSTRAP_SERVERS: "PLAINTEXT://kafka0:19092"
      SCHEMA_REGISTRY_LISTENERS: "http://0.0.0.0:8081"
      SCHEMA_REGISTRY_HOST_NAME: "schema-registry"
      SCHEMA_REGISTRY_KAFKASTORE_TOPIC_REPLICATION_FACTOR: "1"
    depends_on:
    - "kafka0"

  rest-proxy:
    image: confluentinc/cp-kafka-rest:5.2.2
    ports:
    - "8082:8082"
    environment:
      KAFKA_REST_BOOTSTRAP_SERVERS: "PLAINTEXT://kafka0:19092"
      KAFKA_REST_LISTENERS: "http://0.0.0.0:8082/"
      KAFKA_REST_HOST_NAME: "rest-proxy"
      KAFKA_REST_SCHEMA_REGISTRY_URL: "http://schema-registry:8081/"
    depends_on:
    - "kafka0"
    - "schema-registry"

  connect:
    image: confluentinc/cp-kafka-connect:5.2.2
    ports:
```

```
    - "8083:8083"

    environment:

      CONNECT_BOOTSTRAP_SERVERS: "PLAINTEXT://kafka0:19092"

      CONNECT_GROUP_ID: "connect"

      CONNECT_REST_ADVERTISED_HOST_NAME: "connect"

      CONNECT_PLUGIN_PATH: "/usr/share/java"

      CONNECT_INTERNAL_KEY_CONVERTER: "org.apache.kafka.connect.json.JsonConverter"

      CONNECT_INTERNAL_VALUE_CONVERTER: "org.apache.kafka.connect.json.JsonConverter"

      CONNECT_KEY_CONVERTER: "io.confluent.connect.avro.AvroConverter"

      CONNECT_KEY_CONVERTER_SCHEMA_REGISTRY_URL: "http://schema-registry:8081"

      CONNECT_VALUE_CONVERTER: "io.confluent.connect.avro.AvroConverter"


CONNECT_VALUE_CONVERTER_SCHEMA_REGISTRY_URL: "http://schema-registry:8081"

      CONNECT_CONFIG_STORAGE_TOPIC: "connect-config"

      CONNECT_OFFSET_STORAGE_TOPIC: "connect-offset"

      CONNECT_STATUS_STORAGE_TOPIC: "connect-status"
      CONNECT_CONFIG_STORAGE_REPLICATION_FACTOR: "1"
      CONNECT_OFFSET_STORAGE_REPLICATION_FACTOR: "1"
      CONNECT_STATUS_STORAGE_REPLICATION_FACTOR: "1"
    depends_on:
    - "kafka0"
    - "schema-registry"

  ksql:
    image: confluentinc/cp-ksql-server:5.2.2
    ports:
    - "8088:8088"
    environment:
      KSQL_BOOTSTRAP_SERVERS: "PLAINTEXT://kafka0:19092"
      KSQL_LISTENERS: "http://0.0.0.0:8088"
      KSQL_KSQL_SERVICE_ID: "ksql_service_docker"
      KSQL_KSQL_SCHEMA_REGISTRY_URL: "http://schema-registry:8081/"
    depends_on:
    - "kafka0"
    - "schema-registry"
```

```
connect-ui:
  image: landoop/kafka-connect-ui:0.9.7
  ports:
  - "8084:8084"
  environment:
    PORT: "8084"
    PROXY: "true"
    CONNECT_URL: "http://connect:8083"
  depends_on:
  - "connect"

topics-ui:
  image: landoop/kafka-topics-ui:0.9.4
  ports:
  - "8085:8085"
  environment:
    PORT: "8085"
    PROXY: "true"
    KAFKA_REST_PROXY_URL: "http://rest-proxy:8082"
  depends_on:
  - "rest-proxy"
```

```
schema-registry-ui:
  image: landoop/schema-registry-ui:0.9.5
  ports:
  - "8086:8086"
  environment:
    PORT: "8086"
    PROXY: "true"
    SCHEMAREGISTRY_URL: "http://schema-registry:8081/"
  depends_on:
  - "schema-registry"

postgres:
  image: postgres:11
  ports:
  - "5432:5432"
  environment:
    POSTGRES_USER: "cta_admin"
    POSTGRES_PASSWORD: "chicago"
    POSTGRES_DB: "cta"
  volumes:
    - ./producers/data/cta_stations.csv:/tmp/cta_stations.csv
    - ./load_stations.sql:/docker-entrypoint-initdb.d/load_stations.sql
SOURCE FILECREATE TABLE stations (
  stop_id INTEGER PRIMARY KEY,
  direction_id VARCHAR(1) NOT NULL,
  stop_name VARCHAR(70) NOT NULL,
  station_name VARCHAR(70) NOT NULL,
  station_descriptive_name VARCHAR(200) NOT NULL,
  station_id INTEGER NOT NULL,
  "order" INTEGER,
  red BOOLEAN NOT NULL,
  blue BOOLEAN NOT NULL,
  green BOOLEAN NOT NULL
);

COPY stations(
  stop_id,
  direction_id,
  stop_name,
  station_name,
  station_descriptive_name,
  station_id,
  "order",
  red,
  blue,
  green
) FROM '/tmp/cta_stations.csv' DELIMITER ',' CSV HEADER;
```

Connections

FIRST ARDUINO

Red LED – D5

Green LED – D7

Buzzer – A3

Finger print sensor – (UART) – (A1, A2)

TFT Display – (D10, D11, D9, D12, D13)

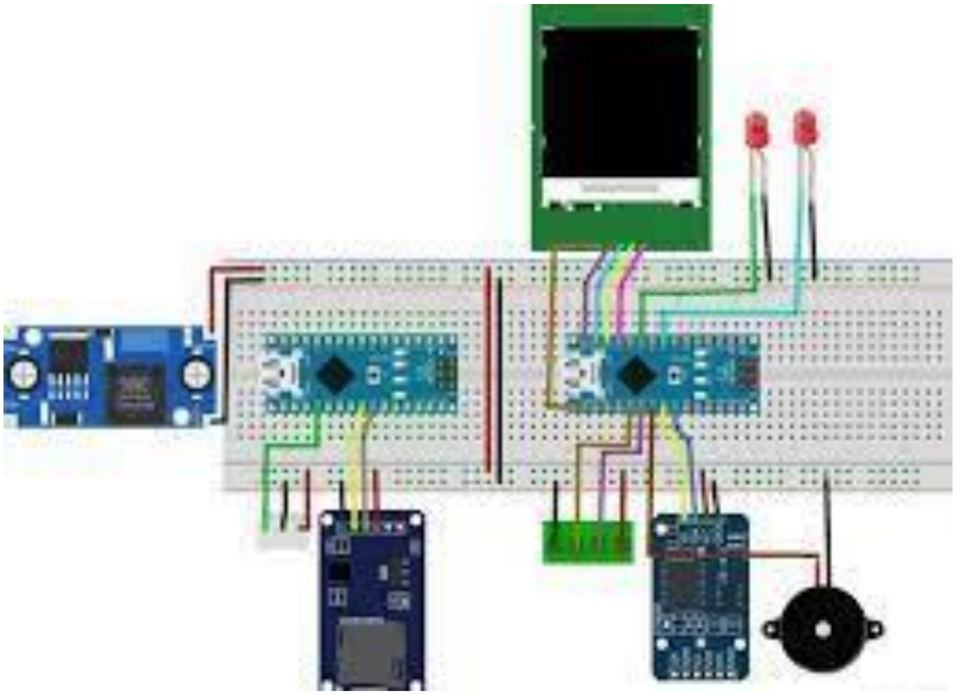
Button – (D4, D2)

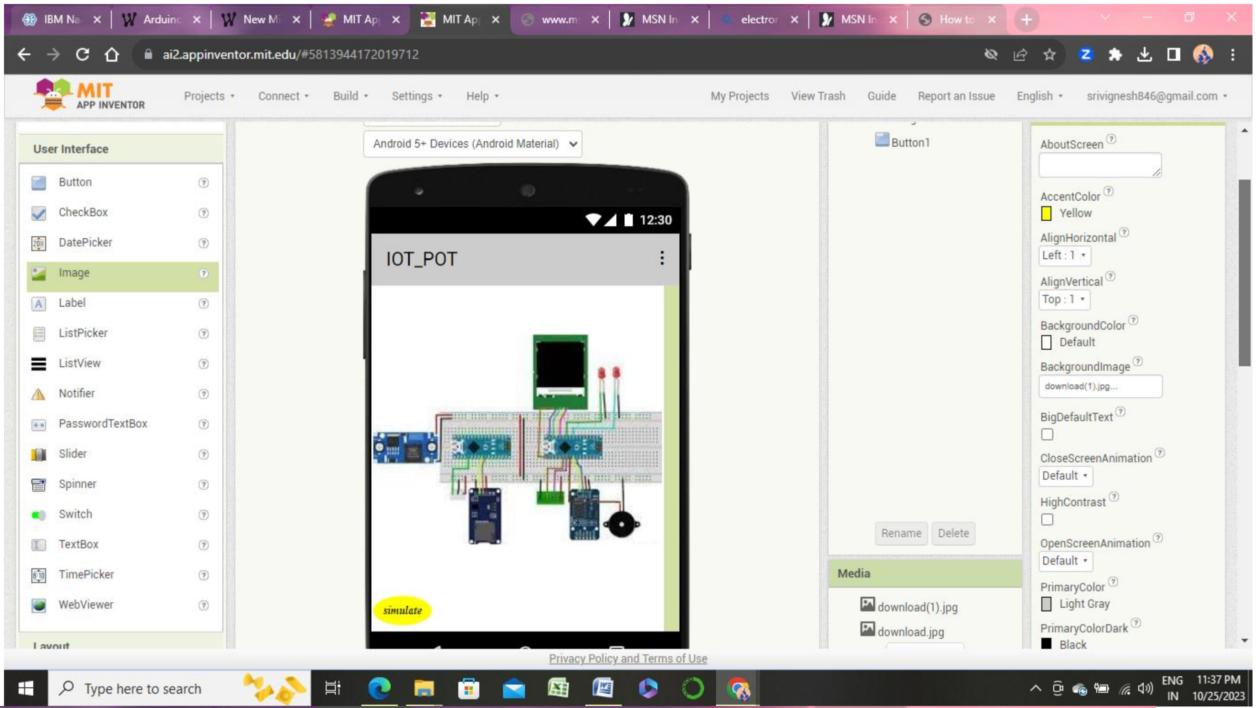
RTC module – I2C- (A4, A5)

SECOND ARDUINO

Hall Effect Sensor – A0

sd card module – SPI – (D10, D11, D12, D13)





AboutScreen

AccentColor: Yellow

AlignHorizontal: Left : 1

AlignVertical: Top : 1

BackgroundColor: Default

BackgroundImage: download(1).jpg...

BigDefaultText: □

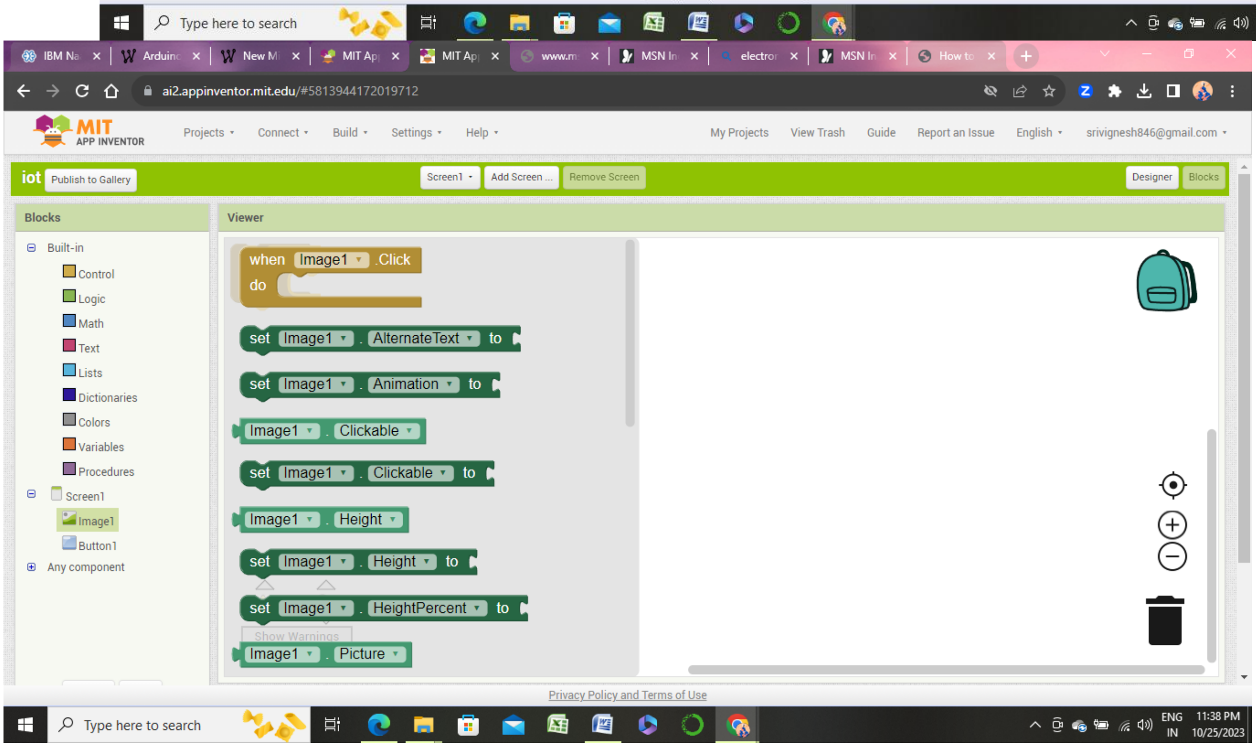
CloseScreenAnimation: Default

HighContrast: □

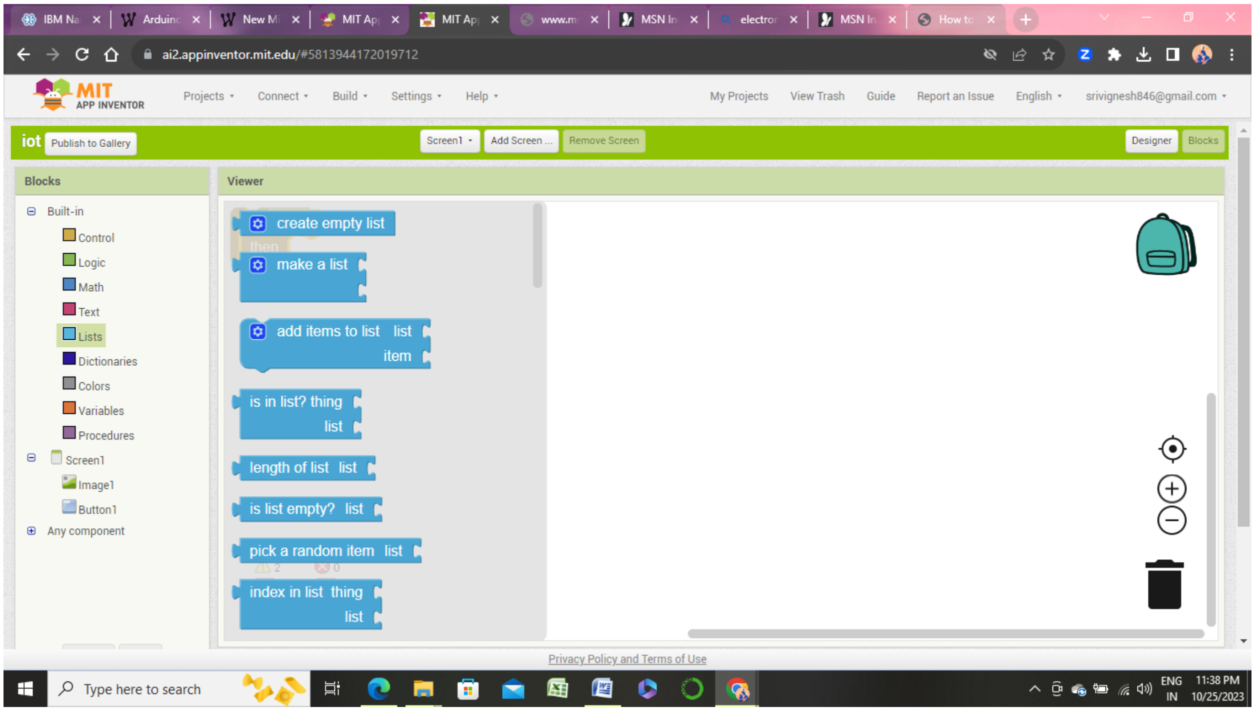
OpenScreenAnimation: Default

PrimaryColor: Light Gray

PrimaryColorDark: Black



```
when Image1 .Click
do

set Image1 . AlternateText to

set Image1 . Animation to

Image1 . Clickable

set Image1 . Clickable to

Image1 . Height

set Image1 . Height to

set Image1 . HeightPercent to

Image1 . Picture
```



CONCLUSION

The above design and works are done by mit web development software in the controllable form we can control it by mobile phones by and app or a website through link