Access Control System

Software Design and Architecture (CSC 419)

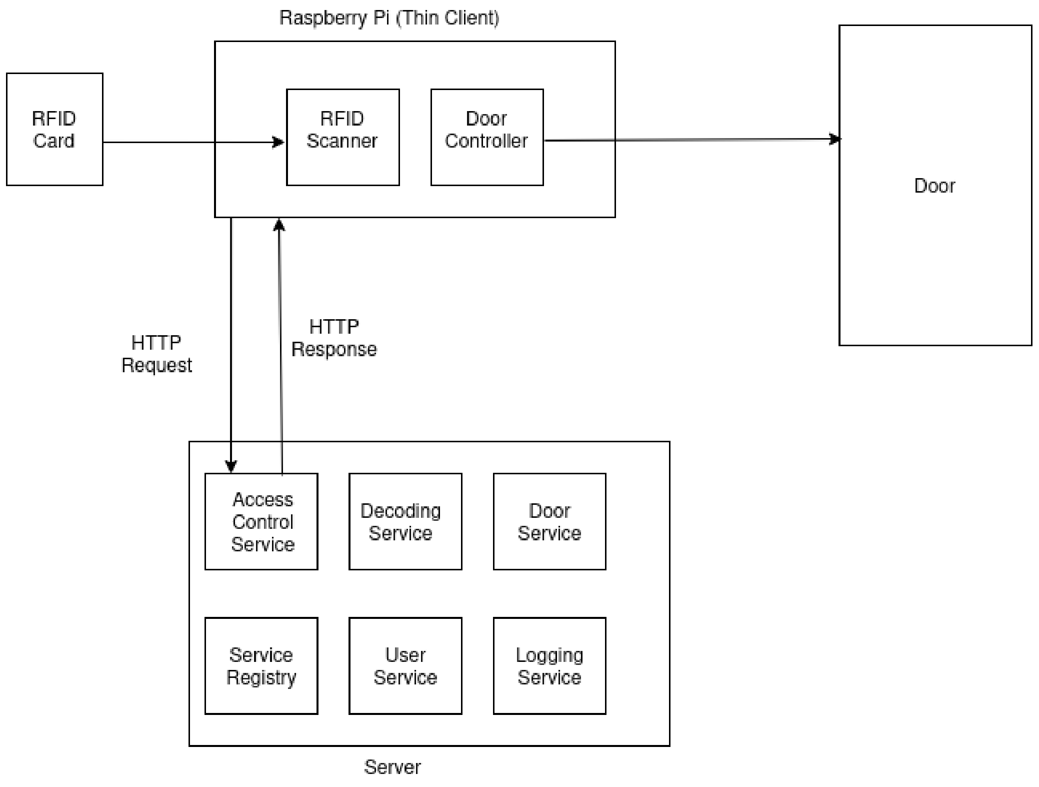
Group 3

1. Olorunfemi-Ojo Daniel Tomiwa	190805503
2. Sanusi Adeyemi Abdulazeez	180805050
3. Muhammed Sodiq Akande	180805056
4. Ezeani David	180805081
5. Ogunrinde Motunrayo Deborah	190805514

Overview

The proposed Access Control System is designed to restrict access to designated spaces using RFID technology integrated with Raspberry Pi devices. This report provides an overview of the system's architecture, including its components, communication protocols, and workflow. The system would consist of the following architectures:

- Event-Driven Architecture
- Client-Server Architecture (Thin Client)
- Service-Oriented Architecture



Event-Driven Architecture

At every door, would be an RFID (Radio Frequency Identification) scanner connected to the GPIO (General-Purpose Input/Output) slot of a Raspberry Pi. The Raspberry Pi continuously listens for events triggered by the RFID scanner. When an RFID card (contains a signature that uniquely identifies an individual) is scanned, an event is detected, and the Raspberry Pi initiates a process to send the scanned data to the Authentication Server. The data sent to the authentication server includes the RFID card signature and the ID of the door trying to be accessed. The hostname of the Raspberry Pi would serve as the Door ID. If access is to be granted, the controller opens the door.

```
import requests
from dnssd import service_discovery
from mfrc522 import SimpleMFRC522

# Door ID
def get_hostname():
    try:
        hostname = socket.gethostname()
        return hostname
    except Exception as e:
        print("An error occurred while getting hostname:", e)


rfid = SimpleMFRC522()
# Get access control service url from registry
url = service_discovery('access-control')

# Listen for data from RFID scanner
while True:
    try:
        id, text = rfid.read()
        payload = {'door': get_hostname(), 'user': text}
        response = requests.post(url, json=payload)
        if response.status_code == 200:
            # Open door

    except:
        continue
```

Client-Server Architecture

The Raspberry Pi acts as a client, sending access request data to the Authentication Server. The server processes the request, performs authentication tasks, and communicates the access decision back to the Raspberry Pi. The Raspberry Pi is a thin client reason being all the processing is done on the server side and therefore, just needs to know whether or not to grant access.

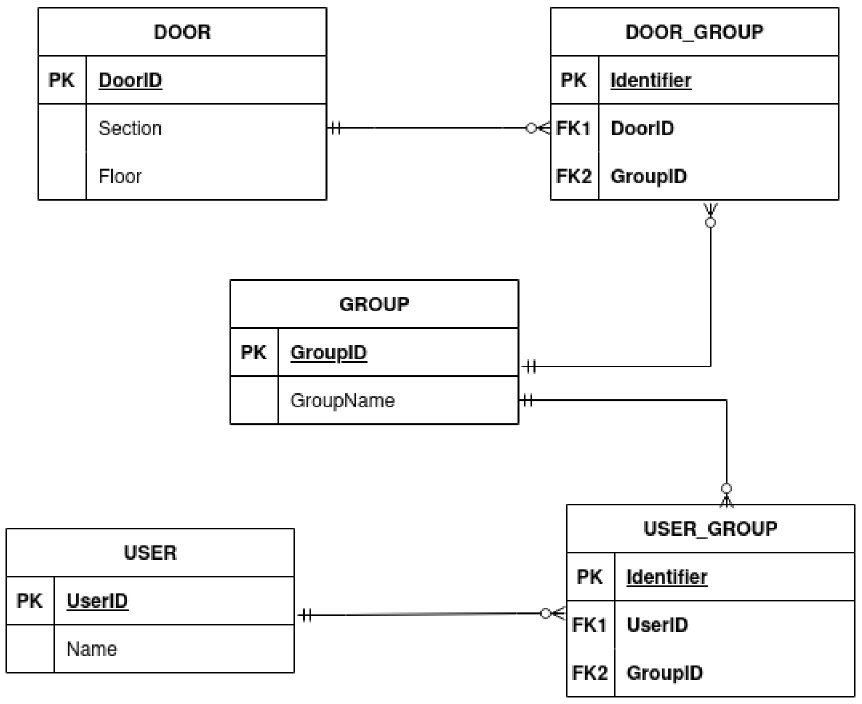
Service-Oriented Architecture

Various services within the Authentication Server collaborate to handle access requests. These services include the Decoding Service, User Service, Door Service, Access Control Service, and Logging Service, each responsible for specific tasks in the authentication process.

1. **Decoding Service:** Decodes the RFID signature retrieved from the scanned data.
2. **User Service:** Fetches user data, including organizational group membership.
3. **Door Service:** Retrieves door-specific data, including organizational group access permissions.
4. **Access Control Service:** Processes access requests, compares user and door group permissions, and grants or denies access accordingly. This service is what the Raspberry Pi interacts with. This service queries a Service Registry to get the domain info with regards to other services and performs the Access Control logic.
5. **Logging Service:** Records access request details and decisions for auditing purposes.

```
// Dummy implementation
```

```
import org.springframework.stereotype.Service;

@Service
public class AccessControlService {

  public HTTPStatus doPost(JSONObject json) {
      String data = deocodeRFIDSignature(json.get("user"));
      User user = fetchUserData(data)
      Door door = fetchDoorData(json.get("hostname"));
      if (door.getPermittedGroups.contains(user.getGroup())) {
         logAccessRequest(...);
         return HTTPStatus.OK;
      }

      logAccessRequest(...);
      return HTTPStatus.UNAUTHORIZED;
  }

  // Log access requests
  public void logAccessRequest(String userId, String doorId, boolean
accessGranted, String timestamp) {
    Service loggingService = getServiceUrlFromRegistry("LoggingService");
    loggingService.send(userId, doorId, accessGranted, timestamp);
  }

  // Decoding Service: Decodes the RFID signature retrieved from the scanned
data
  public String decodeRFIDSignature(String scannedData) {
    Service decodingService = getServiceUrlFromRegistry("DecodingService");
    return decodedService.decode(scannedData);
  }

  // User Service: Fetches user data, including organizational group membership
  public User fetchUserData(String userId) {
    Service userService = getServiceUrlFromRegistry("UserService");
    return userService.get(userId);
  }

  // Door Service: Retrieves door-specific data, including organizational group
access permissions
  public Door fetchDoorData(String doorId) {
    Service doorService = getServiceUrlFromRegistry("DoorService");
    return doorService.get(doorId);
  }

}
```


DOOR
PK DoorID
Section
Floor
DOOR_GROUP
PK Identifier
FK1 DoorID
FK2 GroupID
GROUP
PK GroupID
GroupName
USER
PK UserID
Name
USER_GROUP
PK Identifier
FK1 UserID
FK2 GroupID