Simulation of UE Initial Access and Default Bearer Acquisition

ECE 503 Project 1

Submitted by
Varnith Yemula - A20561527

Faith Anuhya Mallavarupu - A20561526

INDEX

Introduction

❖ **Purpose of the Simulation**:

The primary purpose of this simulation is to demonstrate and analyze the sequential procedures that a User Equipment (UE) undergoes to establish connectivity with a 5G network, from power-on to the acquisition of a default bearer. This involves simulating various key processes defined in the 3GPP TS 123 502 standard, which are critical for the UE's ability to access network services effectively.

❖ **Scope of the Simulation**: Outline the steps covered in the simulation, including the four major steps:

- ➢ Initial Access Procedure
- ➢ Radio Resource Control (RRC) Setup
- ➢ Initial Registration Procedure
- ➢ UE Requested PDU Session Establishment

Simulation Overview

- **Simulation Environment**: We used MATLAB Software to write code and execute the simulation
- **Module Overview**: The main components of the 5G system involved in the simulation,
    - User Equipment (UE)
    - gNodeB (gNB)
    - Access and Mobility Management Function (AMF)
    - Authentication Server Function (AUSF)
    - Session Management Function (SMF)
    - User Plane Function (UPF)
    - Policy Control Function (PCF)
    - Unified Data Management (UDM)
    - Data Network (DN)

Step-by-Step Simulation:

❖ Initial Access Procedure

1) Cell search and selection

A UE may detect signals from numerous cells when it is in the interference region of those cells, which could lead to the receipt of multiple cell IDs. The UE must decode the cell-specific reference signals in order to select a specific cell to camp on. Important information can be obtained from these signals, including channel quality indicators (CQI) that indicate signal strength based on modulation schemes such as QPSK, 16 QAM, 64 QAM, or 256 QAM, and downlink channel estimate for coherent demodulation.

Downlink Channel Estimation for Coherent Demodulation provides information about the characteristics of the downlink channel, which is the wireless communication path from the base station to the user equipment (UE).

The UE usually uses the CQI, which represents the quality of the received signal, to evaluate the reliability and strength of the communication link with the base station.
Depending on the communication system's modulation strategy, it gives a signal strength indication.
There are several modulation techniques that offer different data transmission rates and noise sensitivity, including QPSK, 16 QAM, 64 QAM, and 256 QAM.
The CQI helps the UE understand how strong the received signal is and adapt its communication parameters accordingly, such as adjusting the modulation scheme or power levels to maintain a reliable connection with the base station.

2) System information Reception

There are two main types of blocks used by the cell tower to tell your phone about the network configuration:
Master Information Block (MIB): This shows the detailed information about how to read the SIB message block.

System Information Block (SIB): This provides information about the network configuration. There are different types of SIBs for different purposes.

1) **System Information Block Type 1 (SIB1):** Contains important information like:
   • PLMN Information
   • Tracking Area Code (TAC)
   • Physical Cell ID and specific info

- Scheduling information
- Information about System Information Blocks (SIB2, SIB3, SIB4….) starting with SIB2.

2) **System Information Block 2 (SIB2):** Contains important information like:
   - Common Channel Information
   - Random Access Channel Information
   - Random Access Preamble Info
   - HARQ info

SI messages like SIB1 carry information about the access identities available in a particular cell. Based on the received access identity, the UE determines its configuration. This configuration might involve :

- ❖ Allowed services (voice, data, etc.)
- ❖ Emergency services
- ❖ Public utilities
- ❖ Security services

3) Random Access (RACH) Procedure:

Random Access Channel (RACH) procedure is an essential mechanism in cellular networks, enabling a User Equipment (UE) to communicate with a gNodeB (gNB), the base station.
Steps involved in RACH process are:

1. Random Access Preamble:

The UE's MAC layer generates a special signal called a preamble and sent upwards to the network on the Physical Random Access Channel (PRACH).
The purpose of the preamble is to alert the gNB that a UE wishes to establish a connection.

2. Random Access Response:

After sending the preamble the gNB responds via the downlink shared channel (DL-SCH), specifically using the Physical Downlink Shared Channel (PDSCH).
This response includes:
Cell-Radio Network Temporary Identifier (C-RNTI): A temporary ID assigned to the UE for further communication within the cell, crucial for distinguishing different UEs.
UL-Grant: This is permission and resource allocation for the UE to transmit its next message.

3. RRC Connection Request: Once the UE receives the Random Access Response, it can send an RRC Connection Request message. It is generated by the RRC layer and is the first UE-specific signaling that the UE has transmitted to the gNB which contains the identification of the UE as well as the purpose of the connection request, such as making a call or sending data.

4. MAC Contention Resolution : The last stage deals with the possibility that a number of UEs may have sent RRC Connection Requests at the same time, causing contention on the same channel. These several requests are processed by the gNB, which also selects which UE is granted access to the resources. Subsequently, it notifies the UEs via MAC Contention Resolution message which identifies which UE has connected successfully.

There are some reasons that can lead to the failure of the Random Access Procedure (RACH) in cellular networks like 5G.

- when two or more User Equipments (UEs) send the same preamble at the same time. Given that the preamble is meant to be a unique identifier in the initial access phase, collisions can cause the gNB (gNodeB) to either respond incorrectly or not at all.
- If there are many UEs trying to access the network simultaneously, the available random access opportunities (preambles and slots) might not be sufficient, leading to UEs being unable to send their preamble within the required time.
- If the gNB cannot adequately detect or decode the signal due to poor radio conditions, the connection attempt might fail.

❖ Radio Resource Control (RRC) Setup

RCC set up comes into play once, initial access procedure is successful.

RRCSetupRequest

The first step in RRC setup is **RRCSetupRequest.** It is Initiated By UE by sending a RRCsetup request message to gNB, This message usually contains the information of UE capabilities and the type of services UE wants to use.

Initial UL RRC Message Transfer

The message request sent by UE is sent to gNB-DU , then gNB-DU processes this request based on the current network conditions.
Then gNB-du makes an Initial UL RRC Message Transfer which contains the message request from UE setup request. This is sent to gNB-CU.

The gNB-DU allocates a temporary identifier , called Cell-Radio Network Temporary Identifier (C-RNTI) to UE which is used to identify the UE uniquely within the cell

Process at gNB-CU

The gNB-CU receives the request and allocates the resources needed for UE and configures the RRC connection based on network capacity and UE capability.

gNB-CU generates the RRC setup message which contains RRC connection setup Customized for UE's needs. This message configures the UE for communicating with the network with parameters like security, mobility, and data bearer configurations.

Sending RRC Setup to UE

gNB-CU sends a message to gNB-DU then it is transferred to UE. and UE receives this RRC Setup message and processes the configurations provided by the network.
The UE responds to confirm the completion of setup.

Security Mode Command

After UE sends the RRC setup then the next step is security command mode. gNB sends a security command where UE configures all security related parameters like protocols for encryption and integrity protection.

After UE appling this security protocol it sends a reply SecurityModeComplete mentioning it has completed security command mode.

Context Setup and Configuration

Following the security command the network initiates context setup messages that establish user-specific context in the network. These include assigning resources and capabilities specific to the UE.

Then gNB sends an RRCReconfiguration message.which is used by UE to apply new reconfigurations.

Final Steps for RRC Setup

The gNB-DU sends a UE CONTEXT SETUP RESPONSE message to the gNB-CU to indicate completion of the context setup.
At this point, the network and the UE are ready to engage in data transfer, signaling, or any other required communication.

❖ Initial Registration Procedure

Initiation:

The UE initiates this procedure by sending a Registration Request message. The Registration Request includes vital identifiers like the UE's 5G-GUTI (Globally Unique Temporary Identifier)

The message is transferred through the RAN to the AMF.

Authentication Procedure

When a Registration Request is received by AMF it initiates the authentication process by sending a request to the AUSF. The AUSF communicates with the UDM to get authentication data and generates a response which includes an Authentication Challenge for the UE.Then UE responds to the Authentication Challenge which proves that UE is genuine and this completes the mutual authentication.

Security Mode Command

After successful authentication, the AMF sends a Security Mode Command to the UE. This gives instructions to UE for encryption and decryption process and algorithms for integrity protection for securing all the data and signal transfer. Once UE completes security command it responds with a security command complete message indicating it has completed and applied all security settings.

Permanent equipment Identifier (PEI)

Following that If the permanent equipment identifier (PEI) cannot be retrieved, the AMF can request the UE to provide the PEI by sending the IDENTITY REQUEST message. Then UE will respond by sending IDENTITY RESPONSE including PEI. optionally to verify PEI AMF initiates by calling N5g-eir_EquipmentIdentityCheck_Get service with the N5g-eir_EquipmentIdentityCheck_Get Request message which is sent to the 5G-equipment identity register (5G-EIR).To which 5G-EIR responds back to AMF with PEI check responds via N5g-eir_EquipmentIdentityCheck_Get Response message.

Registration Process

After the initial authentication and security steps, the AMF sends a Nudm_UECM_Registration Request message to the UDM.

The UDM processes the registration request from the AMF and responds with a Nudm_UECM_Registration Response.
AMF Attempts to Retrieve Subscription Data by sending “Nudm_SDM_Get Request” to UDM to get subscription-specific data for the UE.
This information is critical for the AMF to understand what network services are available to the UE and to configure network behavior according to the subscription profile.The AMF sends the REGISTRATION ACCEPT message to the UE indicating that the registration is accepted. This message not only signifies that the registration has been successful but also includes the registration area allocated for the UE

Upon receiving the REGISTRATION ACCEPT message, the UE sends back a REGISTRATION COMPLETE message to the AMF.

❖ UE Requested PDU Session Establishment

Initiation

The UE initiates the process, it sends a NAS message that contains a PDU Session Establishment Request within an N1 SM container, which is encapsulated by the AN in an N2 message to the AMF. This message includes PDU Session ID which is a unique identifier generated by UE.
This request can be “Initial Request”, “Existing Session” or “PDU session Handover”.

AMF Action
Based on the request type, the AMF assesses whether it is a new or existing PDU session and processes accordingly.

AMF Forwards the Request to SMF

if the AMF does not have an existing association with an SMF for the provided PDU session ID, it sends an Nsmf_PDUSession_CreateSMContextRequest to the SMF, forwarding the PDU Session ID along with the N1 SM container received from the UE. The AMF determines the access type and RAT type based on the Global RAN Node ID.

SMF Processes the Request

The SMF interacts with the PCF and UDM to gather necessary information for the PDU session creation. Depending on whether the request is an initial one or a modification, the SMF sends either an N4 Session Establishment or Modification Request to the selected UPF.

From SMF to AMF

The SMF responds to the AMF with either an “Nsmf_PDUSession_CreateSMContext Response” or an “Nsmf_PDUSession_UpdateSMContext Response”, depending on the outcome of the PDU session establishment request.

Optional Secondary Authorization/Authentication

If required, the SMF performs secondary authorization or authentication to further secure the PDU session setup.

SMF Action:

The SMF selects an SSC mode for the PDU session, allocates IP addresses if needed, and establishes policy rules with the PCF for managing the session.

SMF to UPF:

The SMF sends an N4 Session Establishment/Modification Request to the UPF to install packet detection, enforcement, and reporting rules for the session.

SMF to AMF:

The SMF sends the Namf_Communication_N1N2MessageTransfer to the AMF, which includes all session-related information such as PDU Session ID, QoS profiles, and CN Tunnel Info.

AMF to RAN:

The AMF sends the PDU Session Establishment Accept message to the (R)AN, which in turn communicates it to the UE, ensuring that the UE is aware of the session establishment and associated parameters.

Feedback from RAN to AMF:

The RAN sends an N2 PDU Session Response to the AMF, confirming the establishment of the session.

AMF to SMF:

The AMF sends an update request to the SMF to finalize the session context based on the latest information from the (R)AN.

If the PDU Session setup fails at any stage post-initial request, the SMF notifies the AMF of the release and deregisters the session from its management systems.

CODE

```
function main()
   % First Code (Cell Manager and System Information)
   % Cell Manager
   number_of_cells = randi([3, 10]);
   cell_name = randi([17426, 37289]);
   signal_strength = randi([-55, -30]);
   disp(['There are ' num2str(number_of_cells) ' cells available']);
   disp(['Cell ID: ' num2str(cell_name) ' has the highest signal strength
of ' num2str(signal_strength)]);
   disp(['Camped on Cell ID: ' num2str(cell_name)]);
   % System Information
   PLMN_Information = randi([1635, 5937]);
   Tracking_Area_Code = randi([130, 976]);
   Physical_Cell_Id = randi([16357, 45937]);
   Access_Identitiy = randi([0, 14]);
   disp('Received Master Information Block (MIB)');
   disp('Data received with SIB1:');
   disp(['PLMN_Information = ' num2str(PLMN_Information)]);
   disp(['Tracking_Area_Code = ' num2str(Tracking_Area_Code)]);
   disp(['Physical_Cell_Id = ' num2str(Physical_Cell_Id)]);
   prioritize_services(Access_Identitiy);
   % Access Procedure
   C_RNTI = randi([1526, 6954]);
   disp('UE: Sending Random access preamble');
   if check_preamble()
       disp(['gNB: Assigning C-RNTI=' num2str(C_RNTI) ' and UL-GRANT']);
       disp(['UE: Received C-RNTI=' num2str(C_RNTI) ' and UL-GRANT from
gNB']);
       disp('UE: Sending RRC Connection Request');
       disp('gNB: Processing RRC Connection Request from UE');
       if rand() < 0.1
           disp('gNB: Resource allocation failed for UE due to more number
of users');
           disp('Connection establishment failed');
           disp('Initial Access Procedure Failed');
           return;  % Exit the function if setup fails and rest of the code
will not be executed
       else
           disp('gNB: Connection established with UE');
           disp('Initial Access Procedure is Successful');
       end
   else
       disp('Initial Access Procedure Failed');
       return;  % Exit the function if setup fails and rest of the code
will not be executed
   end
   % Second Code (RRCSetupProcess)

   rng('shuffle');
   % NetworkNode functions
   network_congestion = @() rand() < 0.05;
   configuration_mismatch = @() rand() < 0.05;
   system_failure = @() rand() < 0.05;
   % RRCSetupProcess functions
   function setup_success = rrc_setup_request(ue_id, network_node)
       fprintf('UE %d: Sending RRCSetupRequest to gNB-DU.\n', ue_id);
       if network_node.network_congestion()
           fprintf('gNB-DU: Network congestion, unable to process RRC setup
request for UE %d\n', ue_id);
           fprintf('RRC Setup process failed\n');
           setup_success = false;
           return;  % Exit the function if setup fails and rest of the code
will not be executed
       else
           fprintf('gNB-DU: Received RRCSetupRequest from UE %d\n', ue_id);
           setup_success = rrc_setup(ue_id, network_node);
       end
   end
   function rrc_setup_success = rrc_setup(ue_id, network_node)
       fprintf('gNB-DU: Sending RRCSetup to UE %d\n', ue_id);
       fprintf('UE %d: Received RRCSetup message with configuration\n',
ue_id);
       fprintf('gNB-DU: Received RRC CONNECTION SETUP REQUEST from UE
%d\n', ue_id);
       fprintf('gNB-DU: Processing RRC CONNECTION SETUP REQUEST from UE
%d\n', ue_id);
       fprintf('gNB-DU: Sending RRC CONNECTION SETUP COMPLETE to UE %d\n',
ue_id);
       rrc_setup_success = initial_ue_message(ue_id, network_node);
   end
   function initial_ue_message_success = initial_ue_message(ue_id,
network_node)
       fprintf('UE %d: Received RRC CONNECTION SETUP COMPLETE from
gNB-DU\n', ue_id);
```

```
        fprintf('gNB-CU: Received RRC setup complete from UE %d\n', ue_id);
        fprintf('AMF: Received INITIAL UE MESSAGE for UE %d\n', ue_id);
        if network_node.configuration_mismatch()
            fprintf('gNB-CU: Configuration mismatch for UE %d\n', ue_id);
            fprintf('RRC Setup process failed\n');
            initial_ue_message_success = false;
            return;  % Exit the function if setup fails and rest of the code
will not be executed
        else
            initial_ue_message_success = context_setup_request(ue_id,
network_node);
        end
    end
    function context_setup_request_success = context_setup_request(ue_id,
network_node)
        fprintf('gNB-CU: Sending UE CONTEXT SETUP REQUEST for UE %d\n',
ue_id);
        fprintf('gNB-CU: Sending SecurityModeCommand to UE %d\n', ue_id);
        fprintf('gNB-DU: Sending SecurityModeCommand to UE\n');
        fprintf('UE %d: Sending SecurityModeComplete to gNB-DU.\n', ue_id);
        fprintf('gNB-DU: Received SecurityModeComplete from UE\n');
        fprintf('gNB-CU: Initiating RRCReconfiguration for UE %d\n', ue_id);
        fprintf('gNB-DU: Sending RRCReconfiguration to UE %d\n', ue_id);
        fprintf('UE %d: Sending RRCReconfigurationComplete to gNB-DU.\n',
ue_id);
        fprintf('gNB-CU: UE %d setup complete.\n', ue_id);
        context_setup_request_success =
initial_context_setup_response(ue_id, network_node);
    end
    function initial_context_setup_response_success =
initial_context_setup_response(ue_id, network_node)
        if network_node.system_failure()
            fprintf('AMF: System failure, cannot send INITIAL CONTEXT SETUP
RESPONSE for UE %d\n', ue_id);
            initial_context_setup_response_success = false;
            return;  % Exit the function if setup fails and rest of the code
will not be executed
        else
            fprintf('AMF: Sending INITIAL CONTEXT SETUP RESPONSE for UE
%d\n', ue_id);
            initial_context_setup_response_success = true;
        end
    end
```

```
   % Example usage
   network_node = struct('network_congestion', network_congestion, ...
                         'configuration_mismatch', configuration_mismatch,
...
                         'system_failure', system_failure);
   ue_id = randi([2784, 5938]);
   setup_success = rrc_setup_request(ue_id, network_node);
   if setup_success
       fprintf('RRC Setup process successful for UE %d\n', ue_id);
   else
       fprintf('RRC Setup process failed for UE %d\n', ue_id);
       return;  % Exit the function if setup fails and rest of the code
will not be executed
   end
   % Third Code (UE Registration)

   rng('shuffle');
   % NetworkEntity functions
   integrity_check = @() rand() >= 0.05;
   authentication = @() rand() >= 0.05;
   identity_check = @() rand() >= 0.05;
   pei_check = @() rand() >= 0.05;
   % UERegistration functions
   function registration_success = registration_request(ue_id,
network_entity)
       fprintf('UE %d: Sending registration request to the NG-RAN\n',
ue_id);
       fprintf('NG-RAN: Received REGISTRATION REQUEST for UE %d\n', ue_id);

       if ~network_entity.integrity_check()
           fprintf('NG-RAN: Integrity check failed for UE %d\n', ue_id);
           fprintf('Registration process is failed\n');
           registration_success = false;
           return;  % Exit the function if setup fails and rest of the code
will not be executed
       else
           fprintf('NG-RAN: Integrity check is successful, forwarding to
the AMF\n');
           registration_success = amf_registration(ue_id, network_entity);
       end
   end
   function amf_registration_success = amf_registration(ue_id,
network_entity)
```

```
      fprintf('AMF: Authenticating UE %d with AUSF.\n', ue_id);

       if ~network_entity.authentication()
           fprintf('AMF: UE %d authentication is unsuccessful.\n', ue_id);
           fprintf('Registration process is failed\n');
           amf_registration_success = false;
           return;  % Exit the function if setup fails and rest of the code
will not be executed
       else
           fprintf('AMF: UE %d authentication successful.\n', ue_id);
           amf_registration_success = security_mode_command(ue_id,
network_entity);
       end
   end
   function security_mode_command_success = security_mode_command(ue_id,
network_entity)
       fprintf('AMF: Sending SECURITY MODE COMMAND to UE %d\n', ue_id);
       fprintf('UE %d: sending SECURITY MODE COMMAND complete to AMF\n',
ue_id);

       if ~network_entity.identity_check()
           fprintf('AMF : IDENTITY REQUEST message to the UE %d\n', ue_id);
           fprintf('UE %d: Response to IDENTITY REQUEST sending permanent
equipment identifier (PEI)\n', ue_id);
           security_mode_command_success = false;
       else
           security_mode_command_success = pei_registration(ue_id,
network_entity);
       end
   end
   function pei_registration_success = pei_registration(ue_id,
network_entity)
       fprintf('AMF : Initiating PEI check for UE %d from 5G-EIR \n',
ue_id);
       fprintf('5G EIR: Checking PEI identity for UE %d\n', ue_id);

       if ~network_entity.pei_check()
           fprintf('5G-EIR: PEI check failed.\n');
           fprintf('Registration process is failed\n');
           pei_registration_success = false;
           return;  % Exit the function if setup fails and rest of the code
will not be executed
       else
```

```
            fprintf('5G-EIR: PEI check successful\n');
            pei_registration_success = udm_registration(ue_id);
        end
    end
    function udm_registration_success = udm_registration(ue_id)
        fprintf('AMF: Requesting Nudm_UECM_Registration and Nudm_SDM_Get
Request messageto UDM for UE %d\n', ue_id);
        fprintf('UDM: Nudm_UECM_Registration Response and Nudm_SDM_Get
Response message to AMF for UE %d\n', ue_id);
        fprintf('AMF: REGISTRATION ACCEPT message to the UE %d\n', ue_id);
        fprintf('UE %d: REGISTRATION COMPLETE message to the AMF\n', ue_id);
        udm_registration_success = true;
    end
    % Example usage
    network_entity = struct('integrity_check', integrity_check, ...
                            'authentication', authentication, ...
                            'identity_check', identity_check, ...
                            'pei_check', pei_check);
    ue_id = randi([2784, 5938]);
    registration_success = registration_request(ue_id, network_entity);
    if registration_success
        fprintf('Registration process successful for UE %d\n', ue_id);
    else
        fprintf('Registration process failed for UE %d\n', ue_id);
        return;  % Exit the function if setup fails and rest of the code
will not be executed
    end
    % Fourth Code (PDU Session Management)
    session_manager();
    policy_manager(ue_id);
    communication(randi([48276, 71394]));
end
function prioritize_services(Access_Identitiy)
    switch Access_Identitiy
        case 11
            disp('Prioritizing security services');
        case 9
            disp('Prioritizing Emergency services');
        case 10
            disp('Configuring to public utilities');
        otherwise
            disp('Using standard network settings');
    end
```

```
end
function result = check_preamble()
   if rand() < 0.1
       disp('Preamble lost due to poor signal conditions');
       disp('Connection establishment failed');
       disp('Initial Access Procedure Failed');
       result = false;
       return;  % Exit the function if setup fails and rest of the code
will not be executed
   else
       disp('gNB: Received Preamble from UE');
       result = true;
   end
end
function session_manager()
   global ip_address;

   session_states = {'NEW', 'EXISTING'};
   pdu_session_id = randi([48276, 71394]);
   ue_id = randi([2784, 5938]);
   ssc_modes = [1, 2, 3];
   ip_address = {'IPv4', 'IPv6', 'IPv4v6'};
   session_state = session_states{randi([1, numel(session_states)])};
   initiate_session(session_state, pdu_session_id, ue_id, ssc_modes);
end
function initiate_session(session_state, pdu_session_id, ue_id, ssc_modes)
   disp('UE : Initiates PDU Session Establishment for AMF');
   fprintf('AMF: Received PDU Session Establishment request for UE, PDU
Session ID %d.\n and checking for the type of session', pdu_session_id);
   if strcmp(session_state, 'NEW')
       handle_new_session(pdu_session_id, ue_id, ssc_modes);
   elseif strcmp(session_state, 'EXISTING')
       handle_existing_session(ue_id, pdu_session_id);
   end
end
function handle_new_session(pdu_session_id, ue_id, ssc_modes)
   global ip_address;

   disp(['AMF: Handling new session for UE ID ', num2str(ue_id), ', PDU
Session ID ', num2str(pdu_session_id), 'and performing Secondary
authorization/authentication.']);
   if rand() < 0.05
       disp('Authorization/Authentication failed');
```

```
        disp('PDU establishment failed');
        return;  % Exit the function if setup fails and rest of the code
will not be executed
    else
        disp('Authorization/Authentication successful');
        create_sm_context(pdu_session_id, ue_id, ssc_modes);
    end
end
function create_sm_context(pdu_session_id, ue_id, ssc_modes)
    global ip_address;

    disp('AMF: Sending an Nsmf_PDUSession_CreateSMContextRequest to the
SMF');
    if check_subscription_and_policy(ue_id)
        disp(['UPF: Establishing UPF session for PDU Session ID ',
num2str(pdu_session_id), '.']);
        disp(['SMF: PDU Session established successfully for UE ID ',
num2str(ue_id), ' with Session ID ', num2str(pdu_session_id), '.']);
        disp('SMF: Sending an Nsmf_PDUSession_CreateSMContextRequestResponse
to the AMF');
    else
        disp('PDU establishment failed');
        return; % Exit the function if setup fails and rest of the code will
not be executed
    end
end
function handle_existing_session(ue_id, pdu_session_id)
    disp(['AMF: Handling existing session for UE ID ', num2str(ue_id), ',
PDU Session ID ', num2str(pdu_session_id), '.']);
    disp('SMF: Secondary authorization not required for existing session.');
end
function flag = check_subscription_and_policy(ue_id)
    disp(['SMF: Requesting subscription data and policy data for UE ID ',
num2str(ue_id), ' from UDM']);
    subscription_data = logical(randi([0, 1]));
    policy_data = logical(randi([0, 1]));
    flag = subscription_data && policy_data;
end
function policy_manager(ue_id)
    global ip_address;

    selected_ssc_mode = randi([1, 3]);
    selected_ip_address = randi([1, numel(ip_address)]);
```

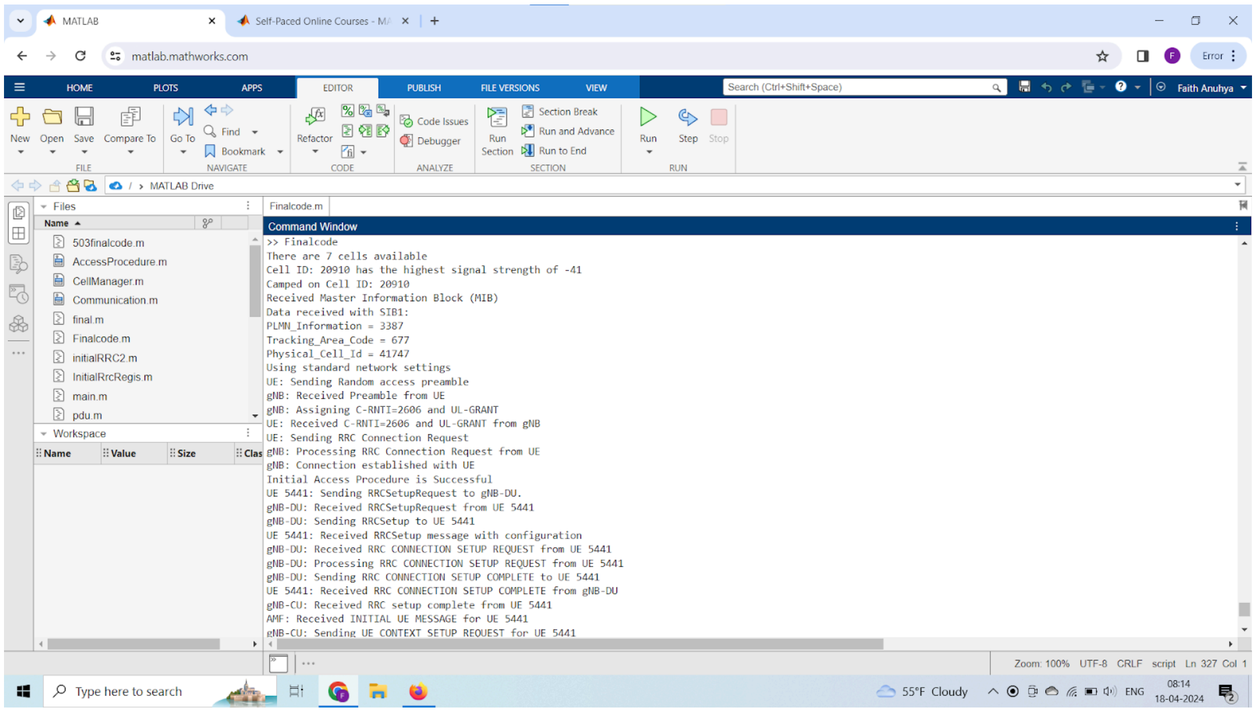
```
   disp('SMF: PCF selected for managing the PDU session.');
   disp('PCF: Providing default PCC rules.');
   disp('SMF: Policy association established with rules');
   disp(['SMF: Selected SSC mode ', num2str(selected_ssc_mode)]);
   disp(['SMF: Selected IP address for ',
ip_address{selected_ip_address}]);
   disp(['SMF: Initiating SM Policy Association Modification for UE ID ',
num2str(ue_id)]);
   disp(['PCF: Modifying policy rules for UE ID ', num2str(ue_id)]);
   disp('PCF: Policy rules updated');
   disp(['SMF: Updated session policy for UE ID ', num2str(ue_id)]);
end
function communication(pdu_session_id)
   global ip_address;

   N1N2_info = struct('PDU_Session_ID', pdu_session_id, 'IP_Address',
ip_address{randi([1, numel(ip_address)])}, 'QoS_Profile', 'Qos_profile',
'CN_Tunnel_Info', 'CN_Tunnel_Info', 'User_Plane_Security_Info',
'User_Plane_Security_Info', 'UE_Integrity_Protection_Max_Data_Rate', '10
Gbps');
   field_names = fieldnames(N1N2_info);
   field_values = struct2cell(N1N2_info);
   info_string = '';
   for i = 1:numel(field_names)
       info_string = [info_string, field_names{i}, ': ',
num2str(field_values{i}), ', '];
   end
   disp('SMF: Namf_Communication_N1N2MessageTransfer to the AMF');
   disp(['AMF: Handling N1N2 Message Transfer with information: ',
info_string]);
   disp(['AMF: Sending NAS message with PDU Session ID ',
num2str(pdu_session_id), ' to RAN.']);
   disp(['RAN: Received NAS message from AMF with PDU Session ID ',
num2str(pdu_session_id), ', allocating N3 Tunnel Info.']);
   disp('RAN: Sending N2 PDU Session Response to AMF with N3 tunnel info');
   disp('SMF: Updating SM context with received N2 SM info');
   disp('UPF: Handling N4 Session Modification request with info');
   disp('UPF: Session modification completed, sending response to SMF.');
   disp('UPF: Delivering buffered down-link packets to UE.');
   disp(['SMF: Sending Update SM Context Response to AMF for PDU Session ID
', num2str(pdu_session_id)]);
   disp(['SMF: Subscribing to mobility events for PDU Session ID ',
num2str(pdu_session_id)]);
```

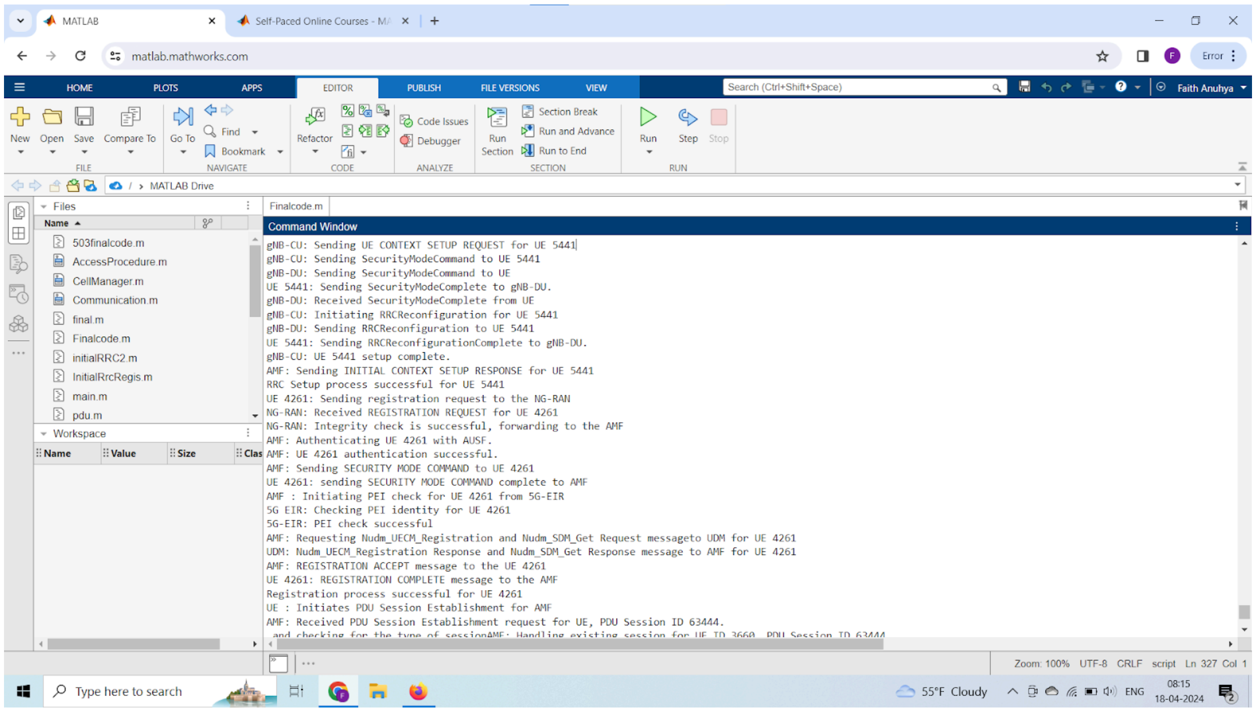
```
   disp(['AMF: Registered for mobility events for PDU Session ID ',
num2str(pdu_session_id)]);
   disp(['SMF: Notifying AMF of SM Context Release for PDU Session ID ',
num2str(pdu_session_id)]);
   disp(['AMF: Handling SM Context Release for PDU Session ID ',
num2str(pdu_session_id)]);
   disp('PDU establishment successful');
end
main();
```

Different Test cases

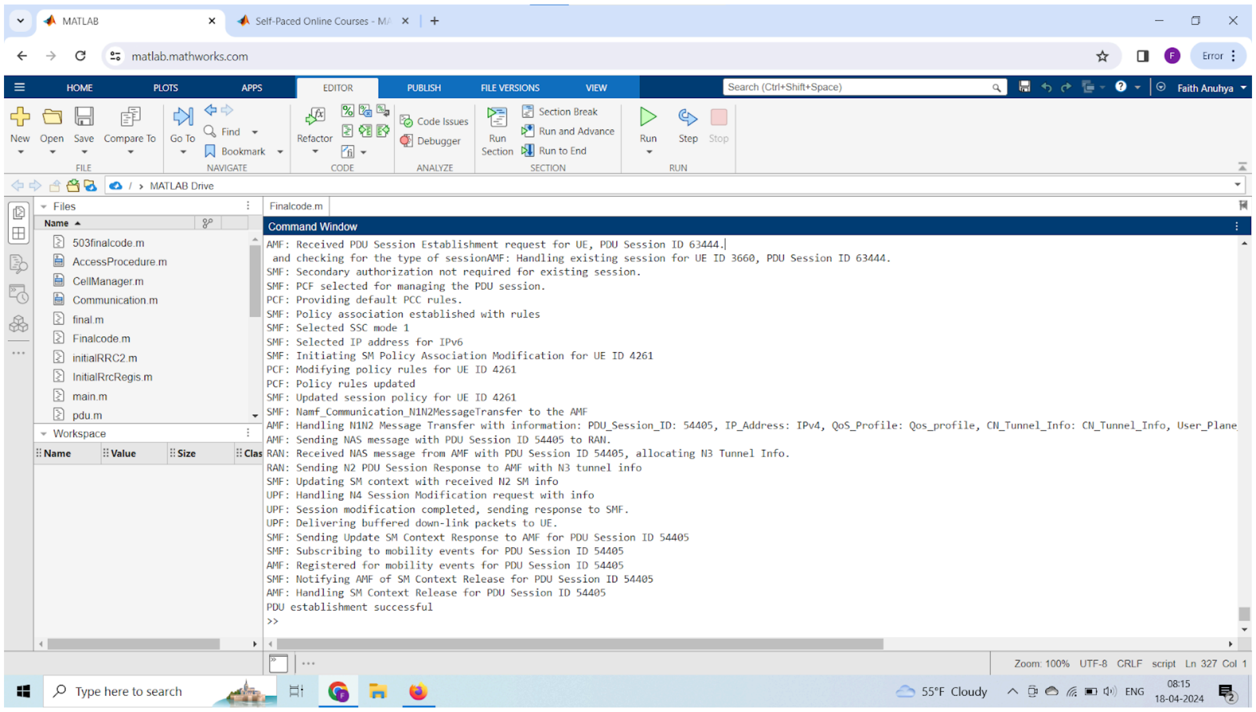
Output if all the steps are successfully implemented:



```
>> Finalcode
There are 7 cells available
Cell ID: 20910 has the highest signal strength of -41
Camped on Cell ID: 20910
Received Master Information Block (MIB)
Data received with SIB1:
PLMN_Information = 3387
Tracking_Area_Code = 677
Physical_Cell_Id = 41747
Using standard network settings
UE: Sending Random access preamble
gNB: Received Preamble from UE
gNB: Assigning C-RNTI=2606 and UL-GRANT
UE: Received C-RNTI=2606 and UL-GRANT from gNB
UE: Sending RRC Connection Request
gNB: Processing RRC Connection Request from UE
gNB: Connection established with UE
Initial Access Procedure is Successful
UE 5441: Sending RRCSetupRequest to gNB-DU.
gNB-DU: Received RRCSetupRequest from UE 5441
gNB-DU: Sending RRCSetup to UE 5441
UE 5441: Received RRCSetup message with configuration
gNB-DU: Received RRC CONNECTION SETUP REQUEST from UE 5441
gNB-DU: Processing RRC CONNECTION SETUP REQUEST from UE 5441
gNB-DU: Sending RRC CONNECTION SETUP COMPLETE to UE 5441
UE 5441: Received RRC CONNECTION SETUP COMPLETE from gNB-DU
gNB-CU: Received RRC setup complete from UE 5441
AMF: Received INITIAL UE MESSAGE for UE 5441
gNB-CU: Sending UE CONTEXT SETUP REQUEST for UE 5441
```

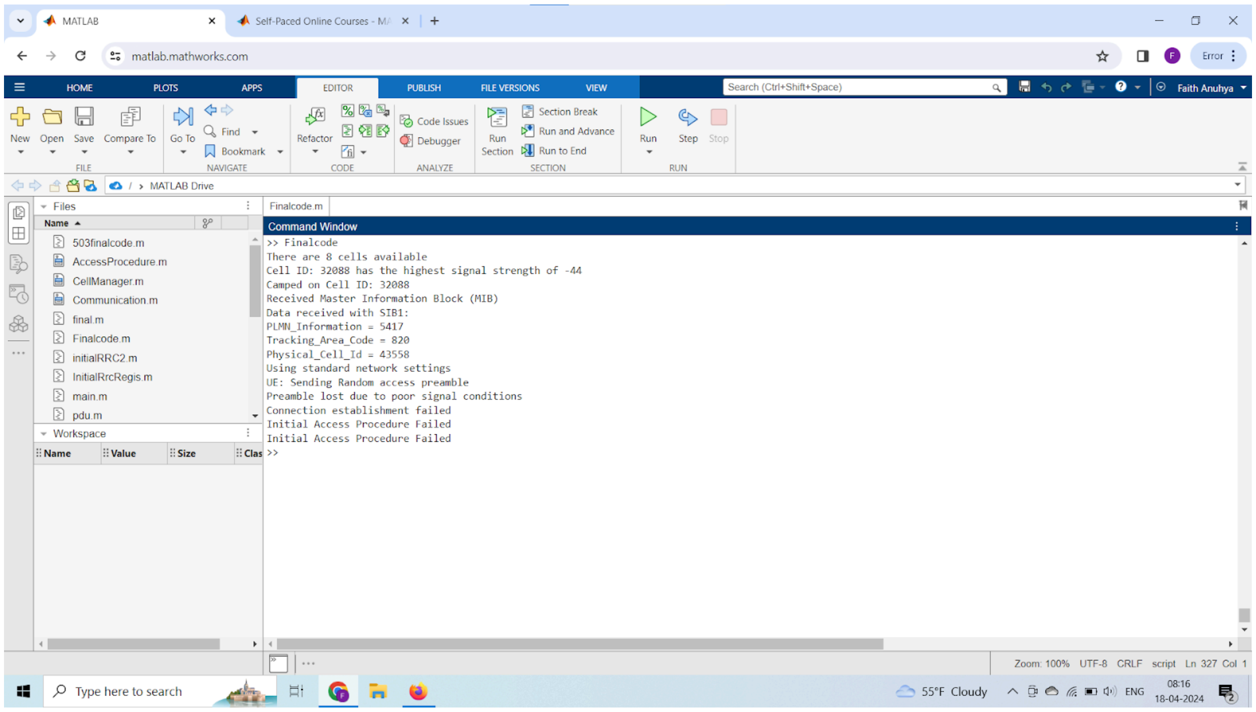


```
gNB-CU: Sending UE CONTEXT SETUP REQUEST for UE 5441
gNB-CU: Sending SecurityModeCommand to UE 5441
gNB-DU: Sending SecurityModeCommand to UE
UE 5441: Sending SecurityModeComplete to gNB-DU.
gNB-DU: Received SecurityModeComplete from UE
gNB-CU: Initiating RRCReconfiguration for UE 5441
gNB-DU: Sending RRCReconfiguration to UE 5441
UE 5441: Sending RRCReconfigurationComplete to gNB-DU.
gNB-CU: UE 5441 setup complete.
AMF: Sending INITIAL CONTEXT SETUP RESPONSE for UE 5441
RRC Setup process successful for UE 5441
UE 4261: Sending registration request to the NG-RAN
NG-RAN: Received REGISTRATION REQUEST for UE 4261
NG-RAN: Integrity check is successful, forwarding to the AMF
AMF: Authenticating UE 4261 with AUSF.
AMF: UE 4261 authentication successful.
AMF: Sending SECURITY MODE COMMAND to UE 4261
UE 4261: sending SECURITY MODE COMMAND complete to AMF
AMF : Initiating PEI check for UE 4261 from 5G-EIR
5G EIR: Checking PEI identity for UE 4261
5G-EIR: PEI check successful
AMF: Requesting Nudm_UECM_Registration and Nudm_SDM_Get Request messageto UDM for UE 4261
UDM: Nudm_UECM_Registration Response and Nudm_SDM_Get Response message to AMF for UE 4261
AMF: REGISTRATION ACCEPT message to the UE 4261
UE 4261: REGISTRATION COMPLETE message to the AMF
Registration process successful for UE 4261
UE : Initiates PDU Session Establishment for AMF
AMF: Received PDU Session Establishment request for UE, PDU Session ID 63444.
 and checking for the type of sessionAMF: Handling existing session for UE ID 3660, PDU Session ID 63444
```



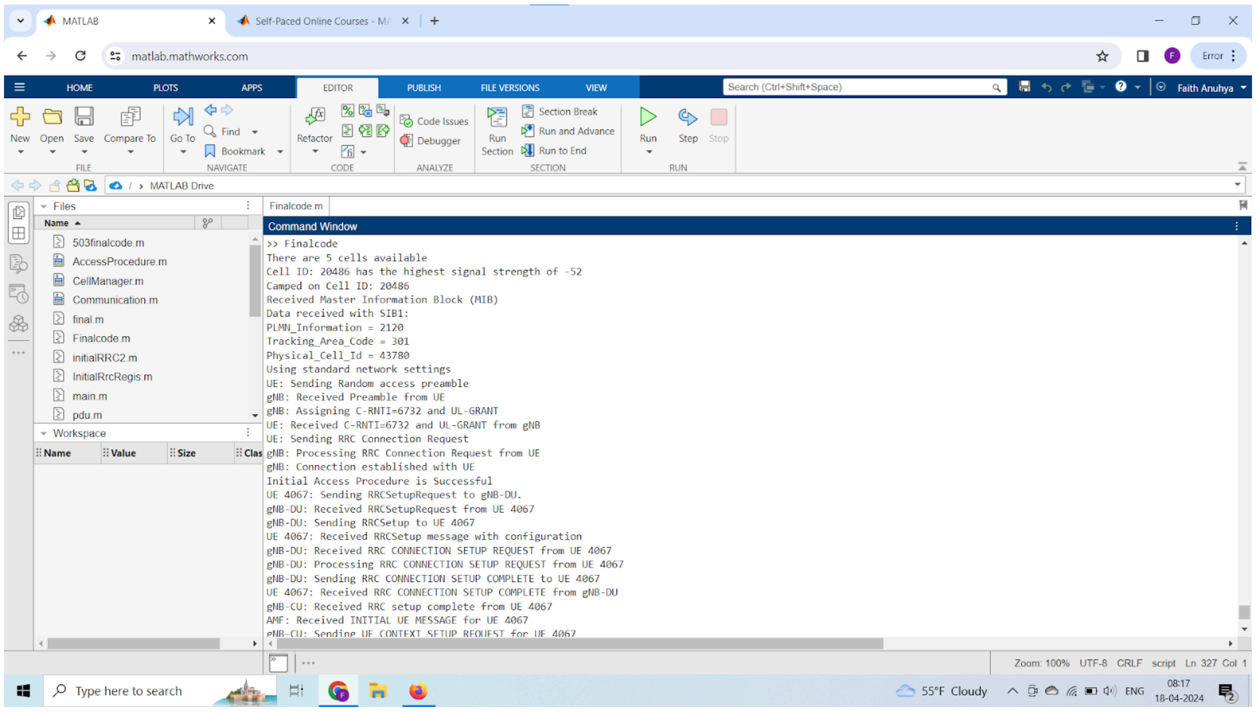
```
AMF: Received PDU Session Establishment request for UE, PDU Session ID 63444.
 and checking for the type of sessionAMF: Handling existing session for UE ID 3660, PDU Session ID 63444.
SMF: Secondary authorization not required for existing session.
SMF: PCF selected for managing the PDU session.
PCF: Providing default PCC rules.
SMF: Policy association established with rules
SMF: Selected SSC mode 1
SMF: Selected IP address for IPv6
SMF: Initiating SM Policy Association Modification for UE ID 4261
PCF: Modifying policy rules for UE ID 4261
PCF: Policy rules updated
SMF: Updated session policy for UE ID 4261
SMF: Namf_Communication_N1N2MessageTransfer to the AMF
AMF: Handling N1N2 Message Transfer with information: PDU_Session_ID: 54405, IP_Address: IPv4, QoS_Profile: Qos_profile, CN_Tunnel_Info: CN_Tunnel_Info, User_Plane
AMF: Sending NAS message with PDU Session ID 54405 to RAN.
RAN: Received NAS message from AMF with PDU Session ID 54405, allocating N3 Tunnel Info.
RAN: Sending N2 PDU Session Response to AMF with N3 tunnel info
SMF: Updating SM context with received N2 SM info
UPF: Handling N4 Session Modification request with info
UPF: Session modification completed, sending response to SMF.
UPF: Delivering buffered down-link packets to UE.
SMF: Sending Update SM Context Response to AMF for PDU Session ID 54405
SMF: Subscribing to mobility events for PDU Session ID 54405
AMF: Registered for mobility events for PDU Session ID 54405
SMF: Notifying AMF of SM Context Release for PDU Session ID 54405
AMF: Handling SM Context Release for PDU Session ID 54405
PDU establishment successful
>>
```

Output if initial access procedure has failed :

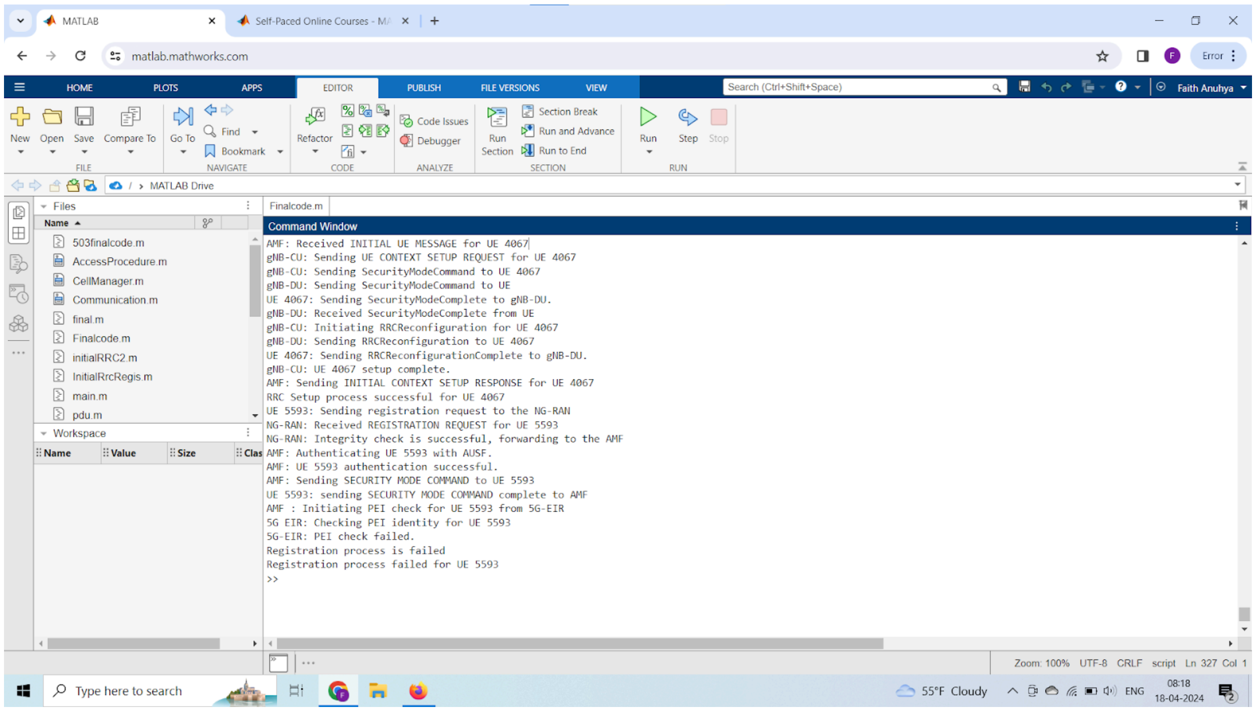


```
>> Finalcode
There are 8 cells available
Cell ID: 32088 has the highest signal strength of -44
Camped on Cell ID: 32088
Received Master Information Block (MIB)
Data received with SIB1:
PLMN_Information = 5417
Tracking_Area_Code = 820
Physical_Cell_Id = 43558
Using standard network settings
UE: Sending Random access preamble
Preamble lost due to poor signal conditions
Connection establishment failed
Initial Access Procedure Failed
Initial Access Procedure Failed
>>
```

Output if the Registration process has failed :

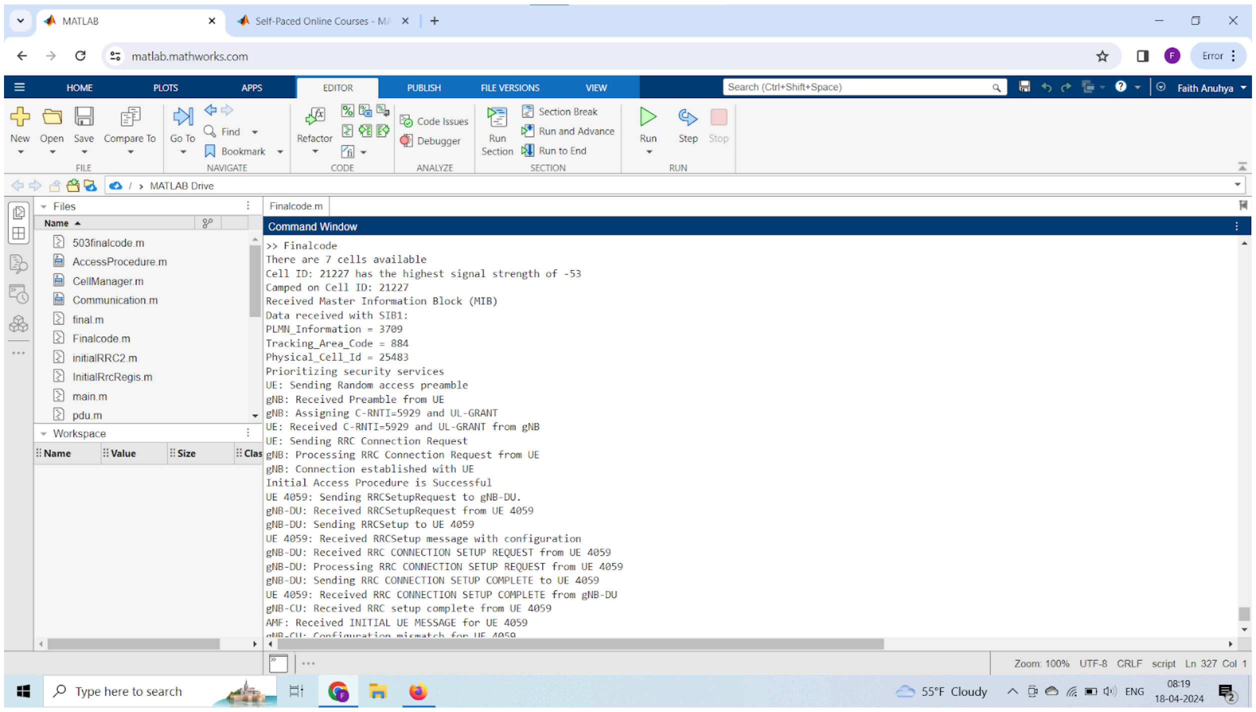


```
>> Finalcode
There are 5 cells available
Cell ID: 20486 has the highest signal strength of -52
Camped on Cell ID: 20486
Received Master Information Block (MIB)
Data received with SIB1:
PLMN_Information = 2120
Tracking_Area_Code = 301
Physical_Cell_Id = 43780
Using standard network settings
UE: Sending Random access preamble
gNB: Received Preamble from UE
gNB: Assigning C-RNTI=6732 and UL-GRANT
UE: Received C-RNTI=6732 and UL-GRANT from gNB
UE: Sending RRC Connection Request
gNB: Processing RRC Connection Request from UE
gNB: Connection established with UE
Initial Access Procedure is Successful
UE 4067: Sending RRCSetupRequest to gNB-DU.
gNB-DU: Received RRCSetupRequest from UE 4067
gNB-DU: Sending RRCSetup to UE 4067
UE 4067: Received RRCSetup message with configuration
gNB-DU: Received RRC CONNECTION SETUP REQUEST from UE 4067
gNB-DU: Processing RRC CONNECTION SETUP REQUEST from UE 4067
gNB-DU: Sending RRC CONNECTION SETUP COMPLETE to UE 4067
UE 4067: Received RRC CONNECTION SETUP COMPLETE from gNB-DU
gNB-CU: Received RRC setup complete from UE 4067
AMF: Received INITIAL UE MESSAGE for UE 4067
gNB-CU: Sending UE CONTEXT SETUP REQUEST for UE 4067
```

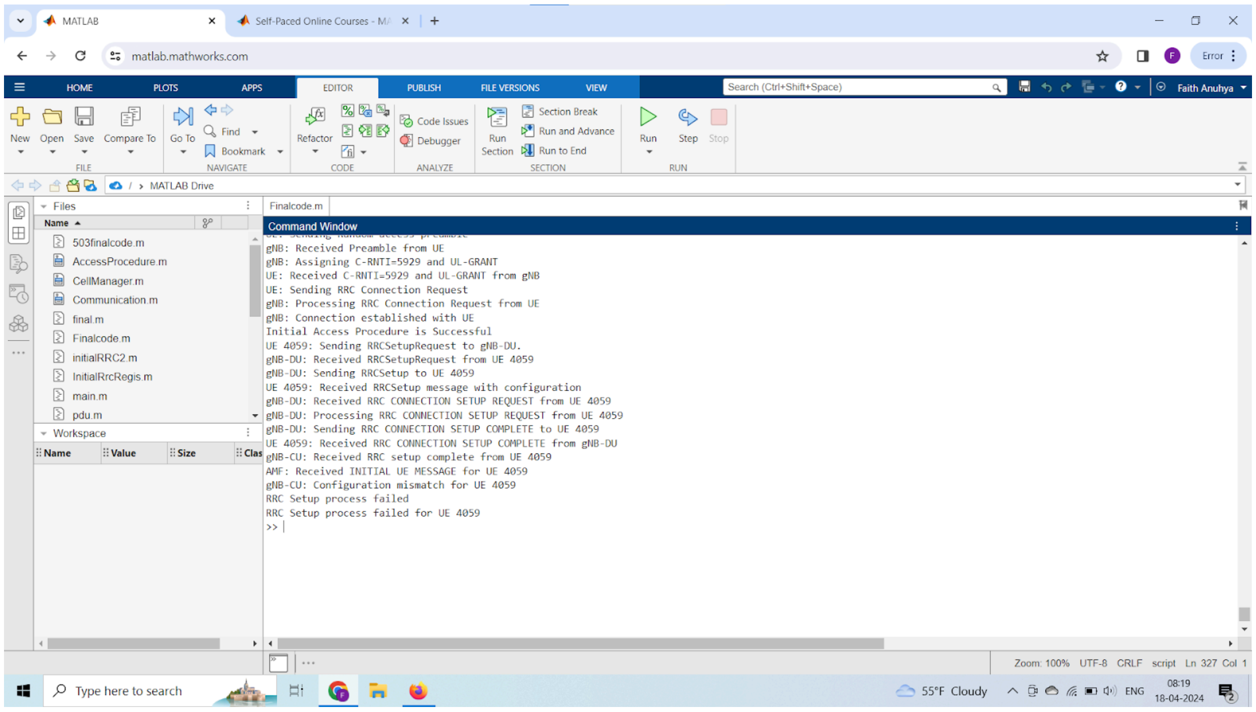


```
AMF: Received INITIAL UE MESSAGE for UE 4067
gNB-CU: Sending UE CONTEXT SETUP REQUEST for UE 4067
gNB-CU: Sending SecurityModeCommand to UE 4067
gNB-DU: Sending SecurityModeCommand to UE
UE 4067: Sending SecurityModeComplete to gNB-DU.
gNB-DU: Received SecurityModeComplete from UE
gNB-CU: Initiating RRCReconfiguration for UE 4067
gNB-DU: Sending RRCReconfiguration to UE 4067
UE 4067: Sending RRCReconfigurationComplete to gNB-DU.
gNB-CU: UE 4067 setup complete.
AMF: Sending INITIAL CONTEXT SETUP RESPONSE for UE 4067
RRC Setup process successful for UE 4067
UE 5593: Sending registration request to the NG-RAN
NG-RAN: Received REGISTRATION REQUEST for UE 5593
NG-RAN: Integrity check is successful, forwarding to the AMF
AMF: Authenticating UE 5593 with AUSF.
AMF: UE 5593 authentication successful.
AMF: Sending SECURITY MODE COMMAND to UE 5593
UE 5593: sending SECURITY MODE COMMAND complete to AMF
AMF : Initiating PEI check for UE 5593 from 5G-EIR
5G EIR: Checking PEI identity for UE 5593
5G-EIR: PEI check failed.
Registration process is failed
Registration process failed for UE 5593
>>
```

Output if the RRC setup has failed :



```
>> Finalcode
There are 7 cells available
Cell ID: 21227 has the highest signal strength of -53
Camped on Cell ID: 21227
Received Master Information Block (MIB)
Data received with SIB1:
PLMN_Information = 3709
Tracking_Area_Code = 884
Physical_Cell_Id = 25483
Prioritizing security services
UE: Sending Random access preamble
gNB: Received Preamble from UE
gNB: Assigning C-RNTI=5929 and UL-GRANT
UE: Received C-RNTI=5929 and UL-GRANT from gNB
UE: Sending RRC Connection Request
gNB: Processing RRC Connection Request from UE
gNB: Connection established with UE
Initial Access Procedure is Successful
UE 4059: Sending RRCSetupRequest to gNB-DU.
gNB-DU: Received RRCSetupRequest from UE 4059
gNB-DU: Sending RRCSetup to UE 4059
UE 4059: Received RRCSetup message with configuration
gNB-DU: Received RRC CONNECTION SETUP REQUEST from UE 4059
gNB-DU: Processing RRC CONNECTION SETUP REQUEST from UE 4059
gNB-DU: Sending RRC CONNECTION SETUP COMPLETE to UE 4059
UE 4059: Received RRC CONNECTION SETUP COMPLETE from gNB-DU
gNB-CU: Received RRC setup complete from UE 4059
AMF: Received INITIAL UE MESSAGE for UE 4059
```



```
gNB: Received Preamble from UE
gNB: Assigning C-RNTI=5929 and UL-GRANT
UE: Received C-RNTI=5929 and UL-GRANT from gNB
UE: Sending RRC Connection Request
gNB: Processing RRC Connection Request from UE
gNB: Connection established with UE
Initial Access Procedure is Successful
UE 4059: Sending RRCSetupRequest to gNB-DU.
gNB-DU: Received RRCSetupRequest from UE 4059
gNB-DU: Sending RRCSetup to UE 4059
UE 4059: Received RRCSetup message with configuration
gNB-DU: Received RRC CONNECTION SETUP REQUEST from UE 4059
gNB-DU: Processing RRC CONNECTION SETUP REQUEST from UE 4059
gNB-DU: Sending RRC CONNECTION SETUP COMPLETE to UE 4059
UE 4059: Received RRC CONNECTION SETUP COMPLETE from gNB-DU
gNB-CU: Received RRC setup complete from UE 4059
AMF: Received INITIAL UE MESSAGE for UE 4059
gNB-CU: Configuration mismatch for UE 4059
RRC Setup process failed
RRC Setup process failed for UE 4059
>>
```

Conclusion

This project effectively simulated the initial access and default bearer acquisition in a 5G network using MATLAB. We explored the detailed steps from powering on the UE to establishing a PDU session, demonstrating the intricate process of network connectivity. The simulation provided insights into the operational protocols of 5G networks and highlighted the importance of each network component.