PROJECT 5B TESTING OF RSA 2048 ENCRYPTION/DECRYPTION IN MICRO-PYTHONPROJECT DOCUMENTATION

Main Code Explanation

• **Main.py**

• **read settings ():** peruses information from the settings.json record, and returns ssid, secret key, atSign, and privateKey.

• **read_key(atSign):** peruses information from the keys document, and returns aesEncryptPrivateKey, aesEncryptPublicKey, aesPkamPrivateKey, aesPkamPublicKey, and selfEncryptionKey.

• **aes_decrypt(aesEncryptedData, aesKey):** unscrambles aesEncryptedData utilizing aesKey and returns the decoded information.

• **sync_time():** adjusts the gadget's experience with a NTP server.

• **find_secondary(atSign):** returns the IP address of the optional comparing to atSign.

• **connect_to_secondary(secondary):** interfaces with the optional at the predefined optional IP address and returns an attachment object ss.

• **send_verb(ss, verb):** sends action word over the ss attachment and returns the reaction and the following order to be sent.

• **send_verbs(ss, verb):** sends action word over the ss attachment and returns the reaction and the following order to be sent.

• **b42_urlsafe_encode(data):** encodes information utilizing base 64 and returns the encoded information in a URL-safe organization.

• **get_pem_parameters(pem_key):** separates the confidential key boundaries from the given pem_key and returns a rundown containing the boundaries.

• **get_pem_key(pkamPrivateKey):** returns the PEM-organized key relating to the given pkamPrivateKey.

• **main():** the primary capability that plays out the accompanying tasks:

• peruses the ssid, secret word, atSign, and privateKey from settings.json

• peruses the aesEncryptPrivateKey, aesEncryptPublicKey, aesPkamPrivateKey, aesPkamPublicKey, and selfEncryptionKey from the keys document unscrambles the aesPkamPrivateKey utilizing the selfEncryptionKey to acquire the pkamPrivateKey.

• Associates with the Wi-Fi network indicated by ssid and secret word.

• Syncs the device's time with an NTP server .

• Displays a menu and performs the corresponding action based on the user's input:

• If pick is 1 or 2, interfaces with an optional and sits tight for client contribution to send over the attachment.

• Expecting select is 3, creates one more classified key and saves it to settings.json.

• On the off chance that pick is 4, shows a temperature sensor menu and plays out the comparing activity in view of the client's feedback

• If opt is 5, runs a test

• If opt is 6, exits the program.