NETWORK PROGRAMMING CA2

Protocol descriptor

TCP Email server

This protocol descriptor showcases the implemented protocol in CA2, it shows the configuration, success and error responses and example exchanges

Cochez Raphael
D00285437@dkit.ie

Summary

Network Programming CA2

Protocol Specification

Configuration:

Host: localhost

Port: 18080

Transmission Encoding: TCP

Data Encoding: JSON-based

Delimiter: %% (double percent sign)

Protocol Command Format

Section	Description
Command	Name of the command (e.g., REGISTER)
Request	Format sent by the client (<COMMAND>%%<JSON Payload>)
Success Response	What the server responds with on success
Error Response	What the server responds with on failure
Example	Full example of client request and server response

REGISTER Protocol Command

Section	Description
Command	REGISTER
Request	REGISTER%%{"email":"<email>", "password":"<password>"}
Success Response	REGISTER_SUCCESS
Error Response	REGISTER_FAIL%%<errorMessage>

Succes Register Example exchange

Client Request: REGISTER%%{"email": "john@example.com", "password": "Secret123"}

Server Response:: REGISTER_SUCCESS

Error Register Example exchange – Email Already Registered

Client Request: REGISTER%%{"email": "john@example.com", "password": "Secret123"}

Server Response:: REGISTER_FAIL%%Email already registered

Error Register Example exchange – Invalid Email Format

Client Request: REGISTER%%{"email": "johexample.com", "password": "Secret123"}

Server Response:: REGISTER_FAIL%%not a valid email address

Error Register Example exchange – Missing Password Field

Client Request: REGISTER%%{"email": "john@example.com"}

Server Response:: REGISTER_FAIL%%Missing required field: password

LOGIN Protocol Command

Section	Description
Command	LOGIN
Request	LOGIN%%{"email":"<email>", "password":"<password>"}
Success Response	LOGIN_SUCCESS
Error Response	LOGIN_FAIL%%<errorMessage>

Succes Login Example exchange

Client Request: LOGIN%%{"email": "john@example.com", "password": "Secret123"}

Server Response:: LOGIN_SUCCESS

Error Example Exchange – Invalid Credentials

Client Request: LOGIN%%{"email": "john@example.com", "password": "Secret123"}

Server Response:: LOGIN_FAIL%%Invalid credentials

Error Example Exchange – Nonexistent User

Client Request: LOGIN%%{"email": "john@example.com", "password": "Secret123"}

Server Response:: LOGIN_FAIL%% User not found

SEND_EMAIL Protocol Command

Section	Description
Command	SEND_EMAIL
Request	SEND_EMAIL%%{"id":"<UUID>","to":"<receiverEmail>", "from":"<senderEmail>","subject":"<string>", "body":"<string>","timestamp":"<timestamp>", "visible":true,"edited":false}
Success Response	SEND_EMAIL_SUCCESS
Error Response	LOGIN_FAIL%%<errorMessage>

Success SEND_EMAIL Example exchange

- **Client Request:** SEND_EMAIL%%{"id":"123","to":"alice@example.com","from":"john@example.com","subject":"Meeting","body":"This is so much writing.","timestamp":"2025-04-11T09:00:00Z","visible":true,"edited":false}
- **Server Response::** SEND_EMAIL_SUCCESS

Error SEND_EMAIL Example exchange – Invalid Recipient

- **Client Request:** SEND_EMAIL%%{"id":"a1b2c3","to":"notregistered@example.com","from":"john@example.com","subject":"Meeting","body":"This is an example ","timestamp":"2025-04-11T09:00:00Z","visible":true,"edited":false}
- **Server Response:** SEND_EMAIL_FAIL%%Recipient not registered

Error SEND_EMAIL Example exchange – Missing Fields

- **Client Request:** SEND_EMAIL%%{"to":"alice@example.com","from":"john@example.com","subject":"Hello","body":"Missing ID and timestamp"}
- **Server Response:** SEND_EMAIL_FAIL%%Missing required fields

RETRIEVE_EMAILS Protocol Command

Section	Description
Command	RETRIEVE_EMAILS
Request	RETRIEVE_EMAILS%%{"email":"<userEmail>", "type":"<sent|received>"}
Success Response	RETRIEVE_EMAILS_SUCCESS%%[<emailJsonObject>, <emailJsonObject>, ...]
Error Response	RETRIEVE_EMAILS_FAIL%%<errorMessage>

Success RETRIEVE_EMAILS Example exchange – Received Emails

Client Request: RETRIEVE_EMAILS%%{"email":"alice@example.com", "type":"received"}

Server Response: RETRIEVE_EMAILS_SUCCESS%%[{"id":"001","from":"bob@example.com","to":"alice@example.com","subject":"Hello","body":"Hi Alice!","timestamp":"2025-04-10T10:00:00Z","visible":true,"edited":false}, {"id":"002","from":"carol@example.com","to":"alice@example.com","subject":"Update","body":"Project status update.","timestamp":"2025-04-10T12:00:00Z","visible":true,"edited":false}]

Success RETRIEVE_EMAILS Example exchange – Sent Emails

Client Request: RETRIEVE_EMAILS%%{"email":"alice@example.com", "type":"sent"}

Server Response: RETRIEVE_EMAILS_SUCCESS%%[{"id":"003","from":"alice@example.com","to":"dave@example.com","subject":"Meeting","body":"See you at 3pm.","timestamp":"2025-04-10T13:00:00Z","visible":true,"edited":false},{"id":"004","from":"alice@example.com","to":"eve@example.com","subject":"Follow-up","body":"Did you get my last email?","timestamp":"2025-04-10T14:00:00Z","visible":true,"edited":false}]

Error RETRIEVE_EMAILS Example exchange – User Not Found

Client Request: RETRIEVE_EMAILS%%{"email":"unknown@example.com"

Server Response: RETRIEVE_EMAILS_FAIL%%User not found

Error RETRIEVE_EMAILS Example exchange – Malformed JSON

Client Request: RETRIEVE_EMAILS%%{"emal":"alice@example.com"}

Server Response: RETRIEVE_EMAILS_FAIL%%Missing or invalid field

READ_EMAIL Protocol Command

Section	Description
Command	READ_EMAIL
Request	READ_EMAIL%%{"email":"<userEmail>", "id":"<emailId>"}
Success Response	READ_EMAIL_SUCCESS%%<emailJsonObject>
Error Response	READ_EMAIL_FAIL%%<errorMessage>

Success READ_EMAILS Example exchange

Client Request: READ_EMAIL%%{"email":"alice@example.com", "id":"001"}

Server Response:
READ_EMAIL_SUCCESS%%{"id":"001","from":"bob@example.com","to":"alice@example.com","subject":"Hello","body":"Hi Alice!","timestamp":"2025-04-10T10:00:00Z","visible":true,"edited":false}

Error READ_EMAIL Example exchange – Email Not Found

Client Request: READ_EMAIL%%{"email":"alice@example.com", "id":"999"}

Server Response: READ_EMAIL_FAIL%%Email not found

Error READ_EMAIL Example exchange – Invalid Request

Client Request: READ_EMAIL%%{"id":"001"}

Server Response: READ_EMAIL_FAIL%%Missing field: email

SEARCH_EMAIL Protocol Command

Section	Description
Command	SEARCH_EMAIL
Request	SEARCH_EMAIL%%{"email":"<userEmail>", "type":"<sent|received>", "keyword":"<searchTerm>"}
Success Response	SEARCH_EMAIL_SUCCESS%%[<emailMetadataObject>, <emailMetadataObject>, ...]
Error Response	SEARCH_EMAIL_FAIL%%<errorMessage>

Success SEARCH_EMAIL Example exchange

- **Client Request:** SEARCH_EMAIL%%{"email":"alice@example.com", "type":"received", "keyword":"project"}
- **Server Response:** SEARCH_EMAIL_SUCCESS%%[{"id":"004","from":"manager@example.com","to":"alice@example.com","subject": "Project deadline","timestamp":"2025-04-10T15:30:00Z"}, {"id":"006","from":"team@example.com","to":"alice@example.com","subject":"Project kickoff","timestamp":"2025-04-11T08:00:00Z"}]

Error SEARCH_EMAIL Example exchange – No Matches

- **Client Request:** SEARCH_EMAIL%%{"email":"alice@example.com", "type":"received", "keyword":"unicorns"}
- **Server Response:** SEARCH_EMAIL_FAIL%%No emails found matching keyword

LOGOUT Protocol Command

Section	Description
Command	LOGOUT
Request	LOGOUT%%{"email":"<userEmail>"}
Success Response	LOGOUT_SUCCESS
Error Response	LOGOUT_FAIL%%<errorMessage>

Success LOGOUT Example exchange

- **Client Request:** LOGOUT%%{"email":"alice@example.com"}
- **Server Response:** LOGOUT_SUCCESS

Error LOGOUT Example exchange – Not Logged In

- **Client Request:** LOGOUT%%{"email":"bob@example.com"}
- **Server Response:** LOGOUT_FAIL%%User not logged in

Error LOGOUT Example exchange – Not Logged In

- **Client Request:** LOGOUT%%{}
- **Server Response:** LOGOUT_FAIL%%Missing required field: email

EXIT_EMAIL Protocol Command

Section	Description
Command	EXIT
Request	EXIT
Success Response	EXIT_SUCCESS
Error Response	*None* — this command is always handled gracefully

Success EXIT Example exchange

- **Client Request:** EXIT
- **Server Response:** EXIT_SUCCESS

Error EXIT Example exchange

The EXIT command allows a client to gracefully terminate the connection. It requires no JSON payload.

Upon receiving EXIT, the server:

- Ends the client's session (if logged in)
- Flushes and saves all in-memory user and email data to disk
- Closes the TCP connection
- Responds with EXIT_SUCCESS before disconnecting