# Chat Application Type Definitions

## Constants

```
#define MAX_USERNAME 32
```

## Message

```
typedef struct Message {
    char *from;              /* sender username (heap allocated) */
    char *text;              /* message content (heap allocated) */
} Message;
```

## Message Vector

```
typedef struct MessageVec {
    Message *data;           /* heap-allocated array */
    int len;                 /* number of messages */
    int cap;                 /* allocated capacity */
} MessageVec;
```

## User

```
typedef struct User {
    char name[MAX_USERNAME]; /* username (fixed buffer) */
    MessageVec inbox;        /* unread messages */
} User;
```

## User Vector

```
typedef struct UserVec {
    User *data;              /* heap-allocated array */
    int len;                 /* number of users */
    int cap;                 /* allocated capacity */
} UserVec;
```

## Server State

```
typedef struct Server {
    UserVec users;           /* all logged-in users */
    int listen_fd;           /* listening socket */
} Server;
```

## RPC Types

```
typedef enum {
    RPC_LOGIN,
    RPC_LOGOUT,
    RPC_TELL,
    RPC_SAY,
    RPC_RECEIVE,
} RpcType;
```

## Request Payloads

```
typedef struct {
    char username[MAX_USERNAME];
} LoginReq;

typedef struct {
    char username[MAX_USERNAME];
} LogoutReq;
```

```
typedef struct {
    char username[MAX_USERNAME];
    char target[MAX_USERNAME];
    char *message;                /* heap allocated */
} TellReq;

typedef struct {
    char username[MAX_USERNAME];
    char *message;                /* heap allocated */
} SayReq;

typedef struct {
    char username[MAX_USERNAME];
} ReceiveReq;
```

## Unified Request

```
typedef struct {
    RpcType type;
    union {
        LoginReq login;
        LogoutReq logout;
        TellReq tell;
        SayReq say;
        ReceiveReq receive;
    };
} Request;
```

## Response Status

```
typedef enum {
    /* success */
    STATUS_OK,

    /* application errors */
    STATUS_ERR_USER_EXISTS,
    STATUS_ERR_USER_NOT_FOUND,
    STATUS_ERR_TARGET_NOT_FOUND,
    STATUS_ERR_MALFORMED,

    /* network error */
    STATUS_ERR_NETWORK,           /* any connection/send/recv failure */
} Status;
```

## Response

```
typedef struct {
    Status status;
    MessageVec messages;          /* only populated for RECEIVE */
} Response;
```