SUCCESS CASE DIAGRAM MariaDB (Data Store) IAuthorizationService (Service Layer) ILoggingService (Service Layer) UserManagementService (Service Layer) UserManagementEntry (Application Layer) UserManagementManager (Business Layer) UserManagementDataAccess (Data Access Layer) SingleOperationRequest(CREATE : String) : String userManagementManager = new UserManagementManager() return UserManagementManager userManagementManager.CreateAccountInput(accountInfo: Map<String, String>): Boolean

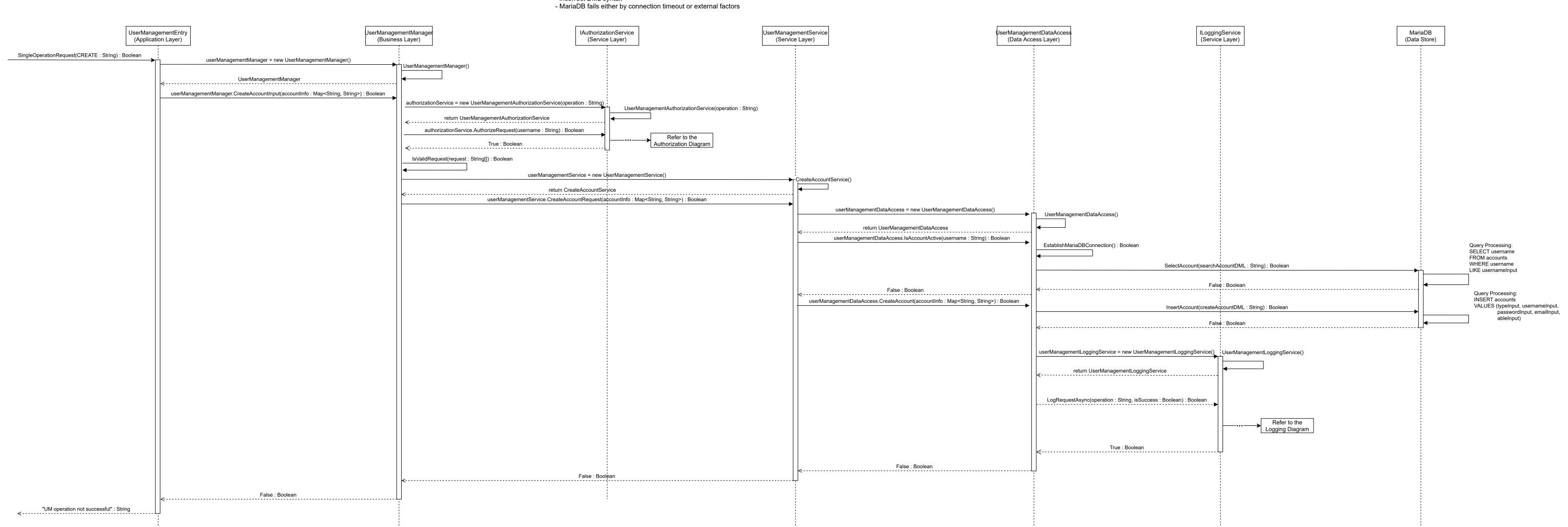
authorizationService = new UserManagementAuthorizationService(operation: String)

UserManagementAuthorizationService(operation: String) return UserManagementAuthorizationService AuthorizeRequest(username : String) : Boolean Refer to the Authorization Diagram True : Boolean IsValidRequest(request : String[]) : Boolean userManagementService = new UserManagementService() UserManagementService() return UserManagementService userManagementService.CreateAccountRequest(accountInfo : Map<String, String>) : Boolean userManagementDataAccess = new UserManagementDataAccess() UserManagementDataAccess() return UserManagementDataAccess userManagementDataAccess.IsAccountActive(username : String) : Boolean

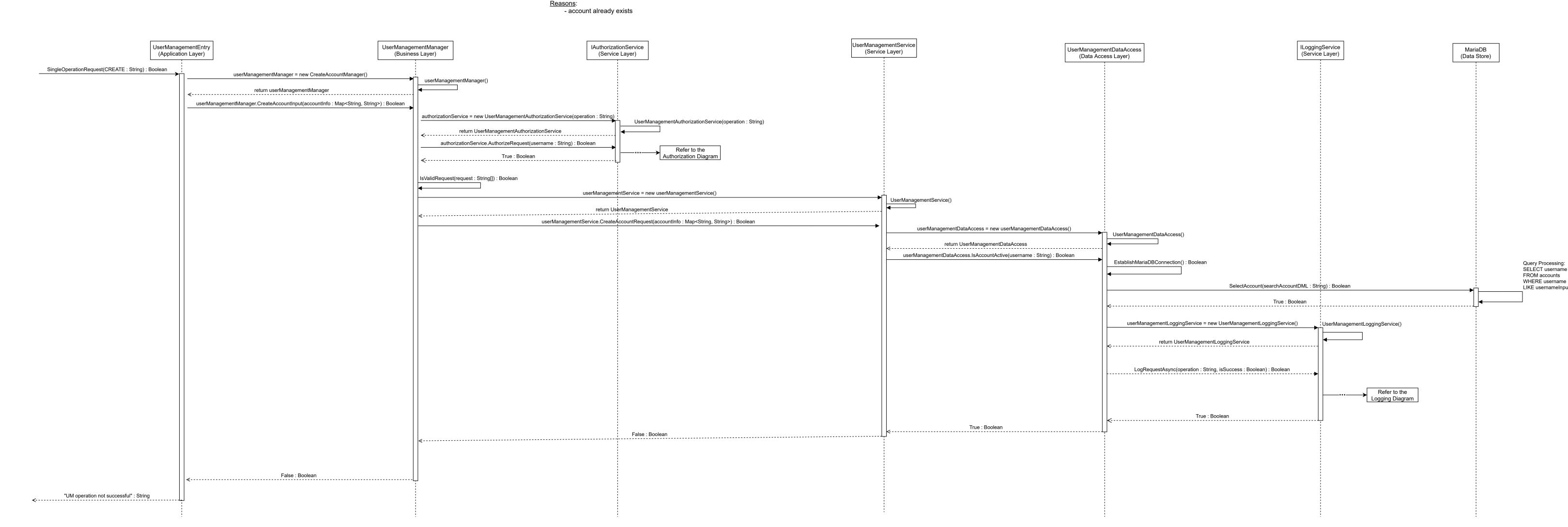
EstablishMariaDBConnection() : Boolean Query Processing: SELECT username FROM accounts WHERE username SelectAccount(searchAccountDML : String) : Boolean LIKE usernameInput False : Boolean False : Boolean Query Processing: INSERT accounts VALUES (typeInput, userManagementDataAccess.CreateAccount(accountInfo : Map<String, String>) : Boolean InsertAccount(createAccountDML : String) : Boolean usernameInput,
passwordInput,
emailInput,
ableInput) True : Boolean True : Boolean True : Boolean userManagementLoggingService = new UserManagementLoggingService() UserManagementLoggingService() return UserManagementLogging\$ervice LogRequestAsync(operation : String, isSuccess : Boolean) : Boolean Refer to the Logging Diagram True : Boolean True : Boolean "UM operation was successful" : String

> **Success Scenario** 1. Single Request is made for the CREATE operation 2. New instance of CreateAccountManager is created Input account info 4. Check if request is valid, the following is checked: - does the input follow the appropriate regex - does the input contain all necessary info for the operation request 5. New instance of CreateAccountService is created 6. Call the CreateAccount Service to process operation with the validated input 7. New instance of CreateAccountDataAccess is created 8. Check if the account already exists using the username 9. Establish the MariaDB connection in DataAccess if not already opened 10. Perform a SELECT query on the Accounts table in the database: SELECT username FROM accounts WHERE username LIKE usernameInput Returns FALSE if not found. 11. Return FALSE to the Service if no account already exists, and continue 12. Create the account with the account info 13. Perform a INSERT query on the Accounts table in the database: INSERT accounts VALUES (typeInput, usernameInput, passwordInput, emailInput, Returns TRUE if account is successfully created. 14. Return TRUE to the Service if the account was successfully created 15. Return TRUE to the Manager if the request was successfully performed 16. Return TRUE to System and follow with the appropriate Logging for this operation

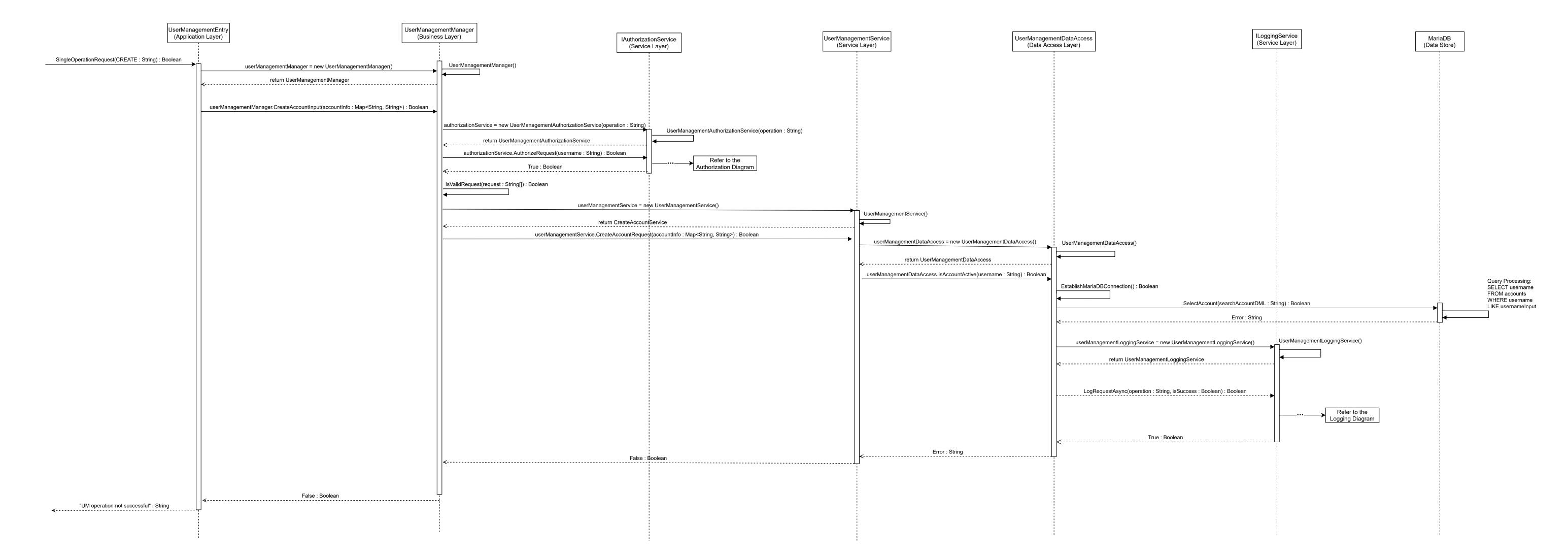
FAILURE CASE DIAGRAMS Fail Case 13 New Account is not Inserted into MariaDB Reasons: - incorrect DML syntax



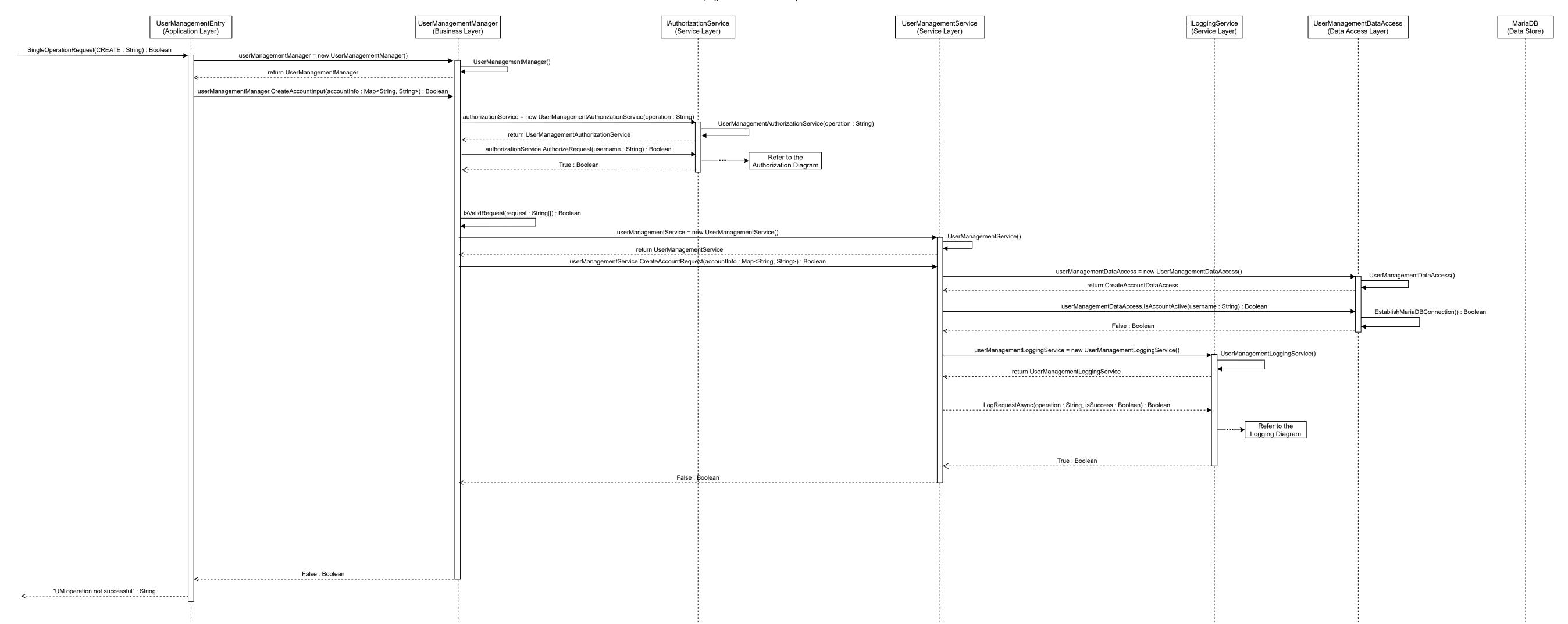
Fail Case 10a New Account already exists in MariaDB



Fail Case 10b Could Not Process Query Reasons: - incorrect DML syntax - MariaDB fails either by connection timeout or external factors



Fail Case 9 MariaDB Connection could not be Established Reasons: - EstablishMariaDBConnection() returns False -- NOTE: continue to call this method until True if still no connection after 5 seconds, log the error and end the process



Fail Case 4 Invalid Request Reasons:

