SUCCESS CASE DIAGRAM ILoggingService (Service Layer) UserManagementDataAccess MariaDB UserManagementService UserManagementEntry (Application Layer) UserManagementManager (Service Layer) (Data Access Layer) (Data Store) (Business Layer) SingleOperationRequest(UPDATE : enum) : Boolean userManagementManager = new UserManagementManager() return UserManagementManager userManagementInput.UserManagementInput(accountInfo : Map<String, String>) : Boolean **Success Scenario** IsValidRequest(request : String[]) : Boolean 1. Single Request is made for the UPDATE operation 2. New instance of UpdateAccountManager is successfully created 3. Input car info return UserManagementService 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 userManagementDataAccess = new UserManagementDataAccess() UserManagementDataAccess() 5. New instance of UpdateAccountService is successfully created return UserManagementDataAccess 6. Call the UpdateAccount Service to process operation with the validated input userManagementService = new UserManagementService() 7. New instance of UpdateAccountDataAccess is successfully created userManagementDataAccess.IsAccountActive(username : String) : Boolean 8. Check if the car already exists using the username
9. Establish the MariaDB connection in DataAccess if not already opened EstablishMariaDBConnection(): Boolean Query Processing: SELECT user 10. Perform a SELECT query on the Accounts table in the database: FROM accounts SELECT car WHERE userName LIKE userNameInput SelectAccount(searchAccountDML : String) : Boolean FROM accounts WHERE car userManagementService.UpdateAccountRequest(accountInfo : Map<String, String>) : Boolean LIKE carInput False : Boolean Returns FALSE if not found. Query Processing: INSERT updatedInfo Make Updates 11. Return FALSE to the Service if no car already exists, and continue userManagementDataAcess.UpdateAccount(accountInfo : Map<String, String>) : Boolean VALUES (userName, email, 12. Create the car with the info given by the user Save Updates userPassword, userCar 13. Perform a INSERT query on the Accounts table in the database: True : Boolean True : Boolean INSERT accounts (Subsequent Logging Process Here) VALUES (typeInput, carYear, carMake, carModel, True : Boolean userManagementLoggingService = new UserManagementLoggingService True : Boolean UserManagementLoggingService() Returns TRUE if car is successfully created and saved to account (Subsequent Logging Process Here) 14. Return TRUE to the Service if the car was successfully created return UserManagementLoggingService 15. Return TRUE to the Manager if the request was successfully performed16. Return TRUE to System and follow with the appropriate Logging for this operation user Management Logging Service. LogRequest Async (UPDATE: String, is Success: Boolean): Booleanlogging diagram True : Boolean SUCCESS CASE DIAGRAM w/ Logging UserManagementEntry (Application Layer) SingleOperationRequest(UPDATE : enum) : String updateAccountManager = new UpdateAccountManager() UserManagementService UserManagementDataAccess (Data Access Layer) MariaDB (Data Store) ILoggingService (Service Layer) return UpdateAccountManager (Service Layer) updateAccountManager.updateAccountInput(accountInfo : Map<String, String>) : Boolean IsValidRequest(request : String[]) : Boolean UMService(operation : String) updateAccountService = new UpdateAccountService() return UpdateAccountService update Account Service. Update Account Request (account Info: Map < String, String >): BooleanupdateAccountDataAccess = new UpdateAccountDataAccess() UMDataAccess(operation : String) return UpdateAccountDataAccess updateAccountDataAccess.IsAccountActive(username : String) : Boolean EstablishMariaDBConnection(): Boolean Query Processing: SELECT user FROM accounts WHERE userName SelectAccount(searchAccountDML : String) : Boolean LIKE userNameInput True : Boolean True : Boolean Query Processing: INSERT updatedInfo updateAccountDataAccess.UpdateAccount(accountInfo : Map<String, String>) : Boolean VALUES (userName, email, InsertAccnt(updateAccountDML : String) : Boolean userPassword, userCar True : Boolean True : Boolean True : Boolean UserManagementLoggingDataAccess(operation : String) userManagementLoggingDataAccess = new UserManagementLoggingDataAccess(CREATE : String) : Boolean EstablishMariaDBConnection(): Boolean Query Processing: INSERT accounts InsertLog(LogSql : String) : Boolean VALUES (timestamp, user, category, description) True : Boolean "UM operation was successful" : String (Subsequent Logging Process Here) Fail Case 3 MariaDB Connection could not be Established - 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 MariaDB (Data Store) UserManagementEntry (Application Layer) UserManagementDataAccess ILoggingService (Service Layer) UserManagementService (Service Layer) (Data Access Layer) UserManagementManager (Business Layer) SingleOperationRequest(UPDATE : enum) : Boolean userManagementManager = new userManagementManager() return UserManagementManager user Management Input (account Info: Map < String, String >): BooleanIsValidRequest(request : String[]) : Boolean userManagementService = new UserManagementService()
return UserManagementService userManagementService.UserManagementRequest(accountInfo : Map<String, String>) : Boolean userManagementDataAccess = new UserManagementDataAccess() userManagementDataAccess.lsAccountActive(username : String) : Boolean return UserManagementDataAccess False : Boolean False : Boolean (Subsequent Logging Process Here) (Subsequent Logging Process Here) False : Boolean (Appropriate Logging Process Here) False : Boolean userManagementLoggingService = new UserManagementLoggingService() return UserManagementLoggingService EstablishMariaDBConnection(): Boofean "UM operation was successful" : String (Subsequent Logging Process Here) userManagementLoggingService.LogRequestAsync(UPDATE : String, isSuccess : Boolean) : Boolean Refer to logging diagram True : Boolean FAILURE CASE DIAGRAMS New Information is not Inserted into MariaDB Reasons: - incorrect DML syntax - MariaDB fails either by connection timeout or external factors MariaDB UserManagementDataAccess (Data Access Layer) ILoggingService (Service Layer) UserManagementManager UserManagementService UserManagementEntry (Data Store) (Application Layer) (Business Layer) (Service Layer) SingleOperationRequest(UPDATE : enum) : Boolean userManagementManager = new UserManagementManager() return UserManagementManager nagementManager.UpdateAccountInput(accountInfo: Map<String, String>): Boolean UserManagementService() userManagementService = new UserManagementService() return UserManagementService userManagementDataAccess = new UserManagementDataAccess() user Management Service. User Management Request (account Info: Map < String, String >): BooleanQuery Processing: SELECT username False : Boolean EstablishMariaDBConnection(): Boolean (Subsequent Logging Process Here) FROM accounts userManagementDataAccess.IsAccountActive(username : String) : Boolean return UserManagementDataAccess WHERE username LIKE usernameInput False : Boolean INSERT accounts VALUES (typeInput, usernameInput, passwordInput, emailInput ableInput) InsertAccount(updaFalse : Boolean: String) : Boolean False : Boolean False : Boolean user Management Data Access. Update Account (account Info: Map < String, String >): Boolean(Subsequent Logging Process Here) False : Boolean userManagementLoggingService = new UserManagementLoggingService() UserManagementLoggingService() False : Boolean return UserManagementLoggingService user Management Logging Service. Log Request Async (UPDATE: String, is Success: Boolean): BooleanRefer to logging diagram True : Boolean Fail Case 2
Could Not Process Query Reasons: - incorrect DML syntax - MariaDB fails either by connection timeout or external factors MariaDB (Data Store) ILoggingService (Service Layer) UserManagementService UserManagementDataAccess UserManagementManager UserManagementEntry (Application Layer) (Data Access Layer) (Business Layer) (Service Layer) SingleOperationRequest(UPDATE : enum) : Boolean userManagementManager = new UserManagementManager() return UserManagementManager IsValidRequest(request : String[]) : Boolean userManagementManager.UserManagementInput(accountInfo : Map<String, String>) : Boolean userManagementService = new UserManagementService() return UserManagementService userManagementService.UserManagementRequest(accountInfo : Map<String, String>) : Boolean userManagementDataAccess = new UserManagementDataAccess() Query Processing: SELECT username return UserManagementDataAccess FROM accounts WHERE username LIKE usernameInput Error : String Error : String False : Boolean False : Boolean (Subsequent Logging Process Here) NOTE: True is the fail safe default so a False : Boolean malicious user does not exploit the False value (Subsequent Logging Process Here) SelectAccount(searchAccountDML : String) : Boolean also we can't represent a database error in a binary result NOTE: True is the fail safe default so a malicious user does not exploit the False value also we can't represent a database error in a binary result userManagementLoggingService = new UserManagementLoggingService() return UserManagementLoggingService "UM operation was successful" : String (Subsequent Logging Process Here) Refer to logging userManagementLoggingService.LogRequestAsync(UPDATE : String, isSuccess : Boolean) : Boolean True : Boolean Fail Case 4 Invalid Request Reasons: - Inputs do not follow their respective regex - Missing required input(s) or does not match the inputs required for request ILoggingService UserManagementService UserManagementDataAccess MariaDB UserManagementManager UserManagementEntry (Business Layer) (Data Access Layer) (Data Store) (Service Layer) (Service Layer) SingleOperationRequest(UPDATE : enum) : Boolean updateAccountManager = new UpdateAccountManager() sValidRequest(request : String[]) : Boolean return UpdateAccountManager False : Boolean update Account Manager. Update Account Input (account Info: Map < String, String >): Boolean(Subsequent Logging Process Here) False : Boolean (Subsequent Logging Process Here) userManagementLoggingService = new UserManagementLoggingService() UserManagementLoggingService() return UserManagementLoggingService userManagementLoggingService.LogRequestAsync(UPDATE : String, is Success : Boolean) : Boolean Refer to logging diagram True : Boolean "UM operation was successful" : String (Subsequent Logging Process Here)

UPDATE Operation