WeCasa

Low Level Design Document

Team HAGS JP

Team Lead:

Allison Austin

Team Members:

Githel Lynn Suico

Haley Nguyen

Joshua Quibin

Judy Li

Matthew Chung

Date Submitted: November 9, 2022

Github Repository:

https://github.com/githelsui/WeCasa

Low Level Design Version Table

Version	Description	Date
1.0	Initial Low Level Design Requirements - Features (Registration, Logging) - Sequence Diagrams - Classes/Interfaces - User Stories	11/9/2022

Table of Contents

Core Components

Registration

Sequence Diagrams

Success Scenarios:

- User registers with a valid email and valid passphrase. The system is able to assign a system-wide unique username. A system message displays “Account created successfully” within 5 seconds of invoking registration process. The system provides the username to the user.

Failure Scenarios:

- User registers with an invalid email. A system message displays “Invalid email provided. Retry again or contact system administrator” or no system message. Account is not created.
- User registers with an invalid passphrase. A system message displays “Invalid passphrase provided. Retry again or contact system administrator” or no system message, Account is not created.
- User registers with a valid email and valid passphrase. The system was unable to assign a system-wide username. A system message displays “Unable to assign username. Retry again or contact system administrator”. Account is not created.
- User registers with a valid email and valid passphrase. The system was able to assign a system-wide username. Entire process took longer than 5 seconds. A system log entry is recorded. Account is created.

Classes/Interface

- IUserManager
    - public bool ValidateEmail(string email);
    - public Result ValidatePassword(string Password);
    - public string ConfirmPassword(string password);
    - public Result RegisterUser(string email, string password);
- UserManager extends IUserManager
    - public bool ValidateEmail(string email)
    - public Result ValidatePassword(string password)
    - public string ConfirmPassword(string password)
    - public Result RegisterUser(string email, string password)
- UserCreationIntegrationTests
    - public void ShouldCreateWithin5Seconds()
- UserCreationUnitTests

    - public void ShouldHaveValidPasswordLength()
    - public void ShouldHaveValidPasswordCharacters()
    - public void ShouldHaveValidEmailLength()
    - public void ShouldHaveValidEmail()

User Stories

- As a user, I want to successfully register an account with a valid email and password and receive a confirmation message from the system within 5 seconds after registering.
- As a developer, I want users to be redirected to a system administrator in the event that the registration process fails.
- As a developer, I want to validate a user's inputted email address to ensure it is a viable email that is not linked to any pre-existing registered accounts.
- As a developer, I want to validate a user's inputted password based on the following criteria:
    - Blank Spaces
    - Lowercase Characters: a-z (*)
    - Uppercase Characters: A-Z (*)
    - Numbers: 0-9 (*)
    - Special Character: .,@!- (*)

  (*) Users must include at least one of the specified character type within their password

Logging

Sequence Diagrams

Success Scenarios:

- The system logs system success events
- The system logs system failure events
- The system logs user success events
- The system logs user failure events

Failure Scenarios:

- The logging process took longer than 5 seconds to complete upon invocation
- The logging process blocks a user from interacting with the system
- The logging process completes within 5 seconds, but did not save to a persistent data store
- The logging process completes within 5 seconds, but did not accurately save the event to the persistent data store (i.e. timestamp, log level, category, message, etc.)
- Previously saved log entries are modifiable

Classes/Interface

- ILogger
    - Log(sqlObject)
- UserManagementLogger extends ILogger
- GroupLogger extends ILogger
- NotificationLogger extends ILogger
- SQLObject
    - Variables: connection, sql statement,
    - Execute()

User Stories

- As a developer, I want to log flow of user operations, tracking key user information, user events that may lead to system failure, and system errors.
- As a developer, I want to log flow of system operations, tracking key system information, system events that may lead to system failure, and system errors.