Design Document: Exogredient

Team A

Jason Nguyen (Team Leader) - 014232658
Eleazar Gomez - 012465490
David Nguyen - 015633187
Abe Malla - 016069649
Charles Kwak - 015368728
Hovhannes Babuchyan - 017228651

November 12, 2019

Table of Contents

Features **4**

Upload - Create An Upload 4

Upload - Save Progress Of An Upload 4

Upload - Delete Progress Of An Upload 5

Upload - Continue Progress Of An Upload 6

Upload - Delete Own Upload 6

Report An Upload 7

Search For An Ingredient 7

Comment On An Upload 8

Search For A Store 9

Claim Store 9

Upvote / Downvote (Upload Or Store Owner’s Description) 10

Display Profile View 11

Admin Delete Upload 11

Display Manage Store View. 12

Manage Store - Flag / Unflag An Ingredient In Their Store 12

Manage Store - Advertisement 13

Manage Store - Submit Description Of An Ingredient 13

Manage Store - Unclaim Store 14

Verify An Upload 15

Ingredient Save List - Add To Save List 15

Ingredient Save List - Delete From Save List 16

Ingredient Save List - View Item From Save List 16

Registration 17

Authentication - Login 18

Authentication - Logout 19

Authentication - Reset Password 19

Authorization 21

Logging 21

Archiving 21

Error Handling 22

Low-Level Diagrams **23**

Logging Low Level 23

Authentication Low Level 24

Authorization Low Level 25

Archiving Low Level 26

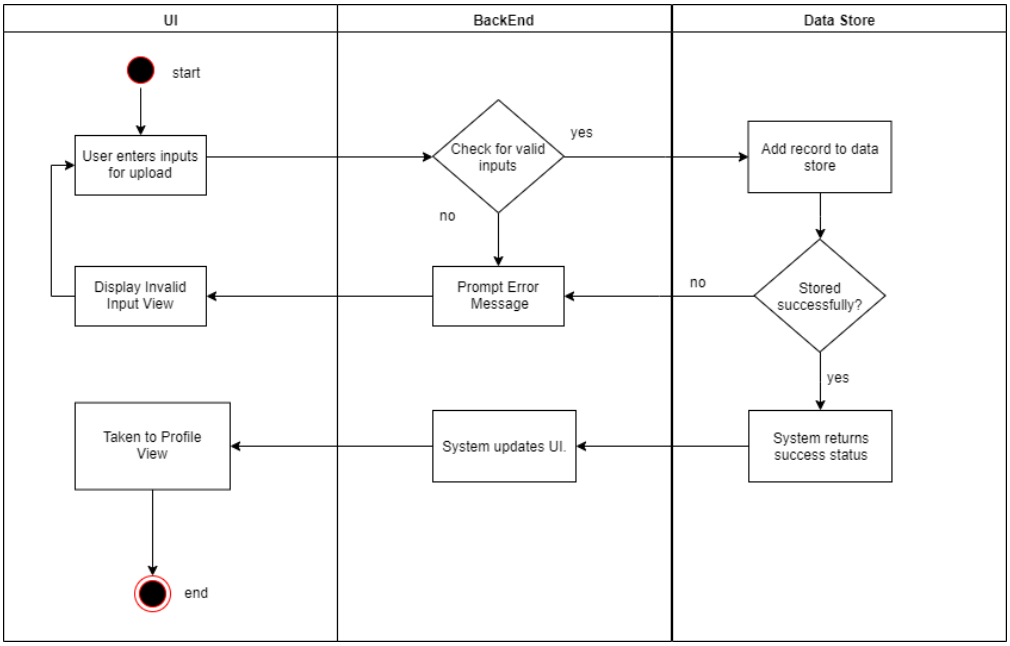
Error Handling Low Level 27
Registration Low Level 28
Reset Password Low Level 30
Code Validation Low Level 31

Features

Upload - Create An Upload

Preconditions:

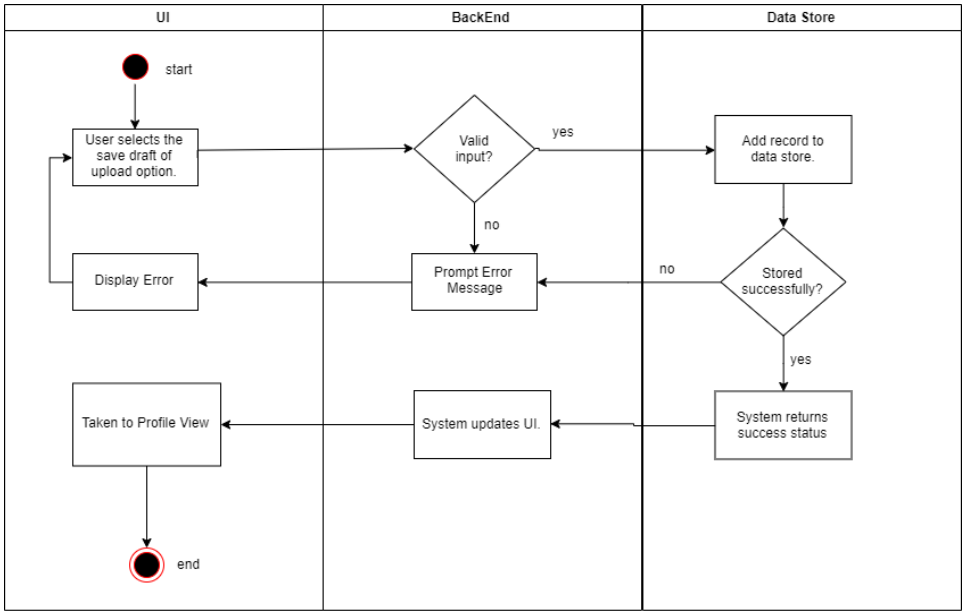
- Must be logged in.
- Must be at the Upload View.



Upload - Save Progress Of An Upload

Preconditions:

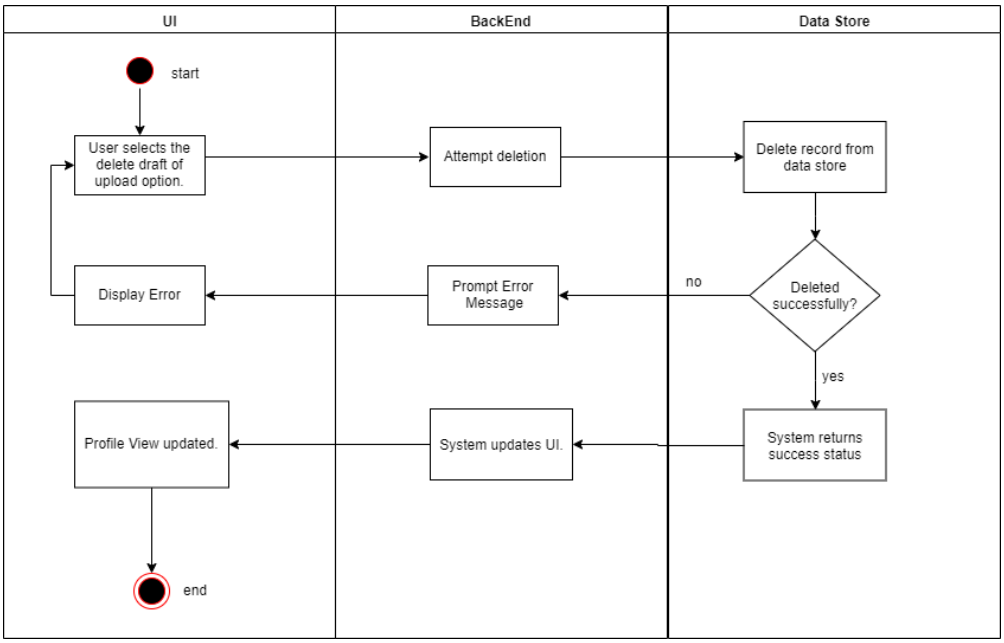
- Must be logged in.
- Must be at the Upload View.



Upload - Delete Progress Of An Upload

Preconditions:

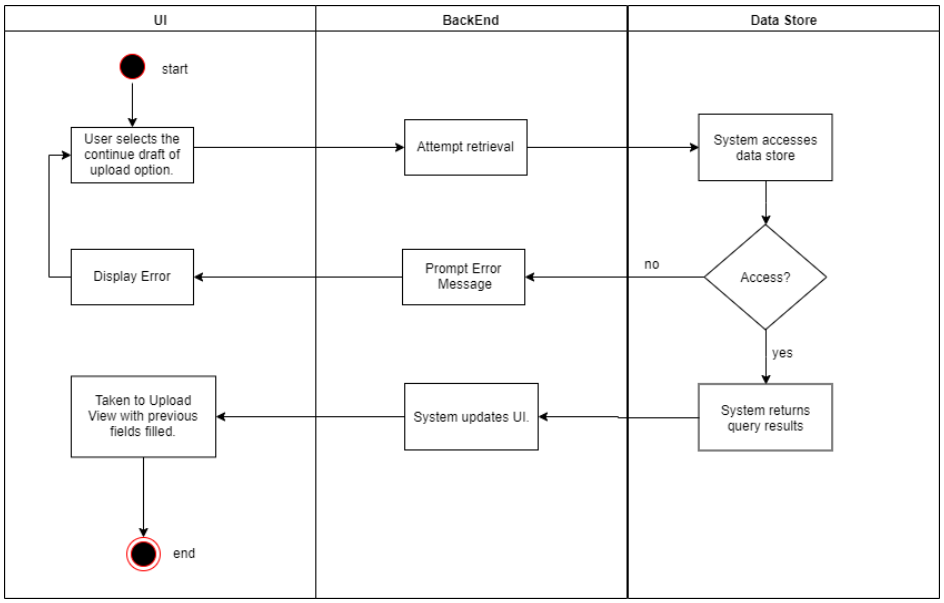
- Must be logged in.
- Must have a saved Upload draft.
- Must be at the Profile View.



Upload - Continue Progress Of An Upload

Preconditions:

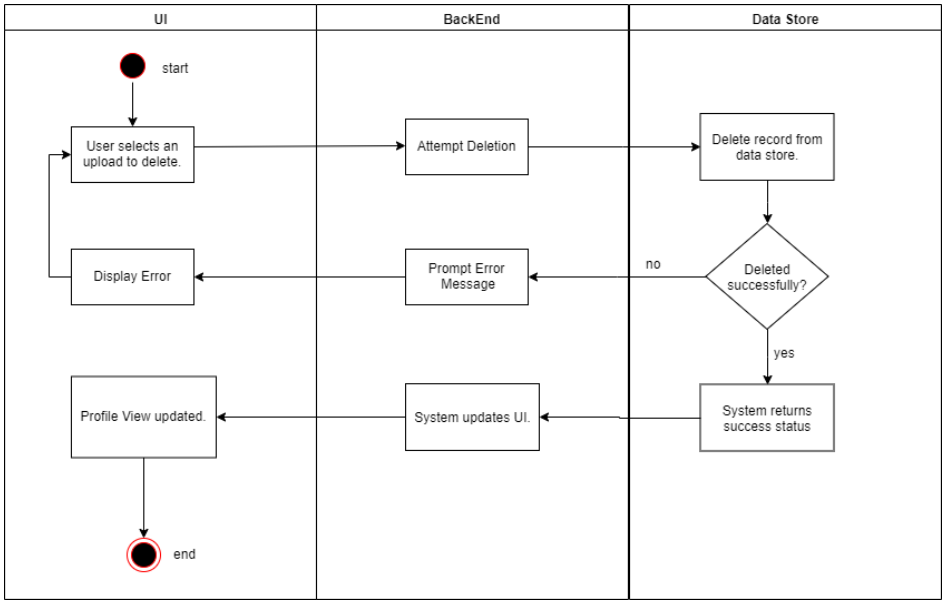
- Must be logged in.
- Must have a saved Upload draft.
- Must be at the Profile View.



Upload - Delete Own Upload

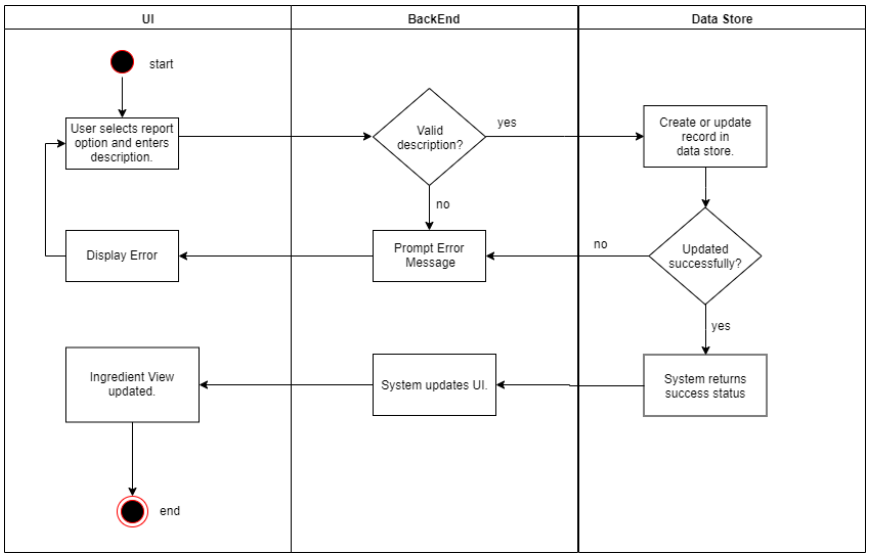
Preconditions:

- Must be logged in.
- Must be at the Profile View.



Report An Upload

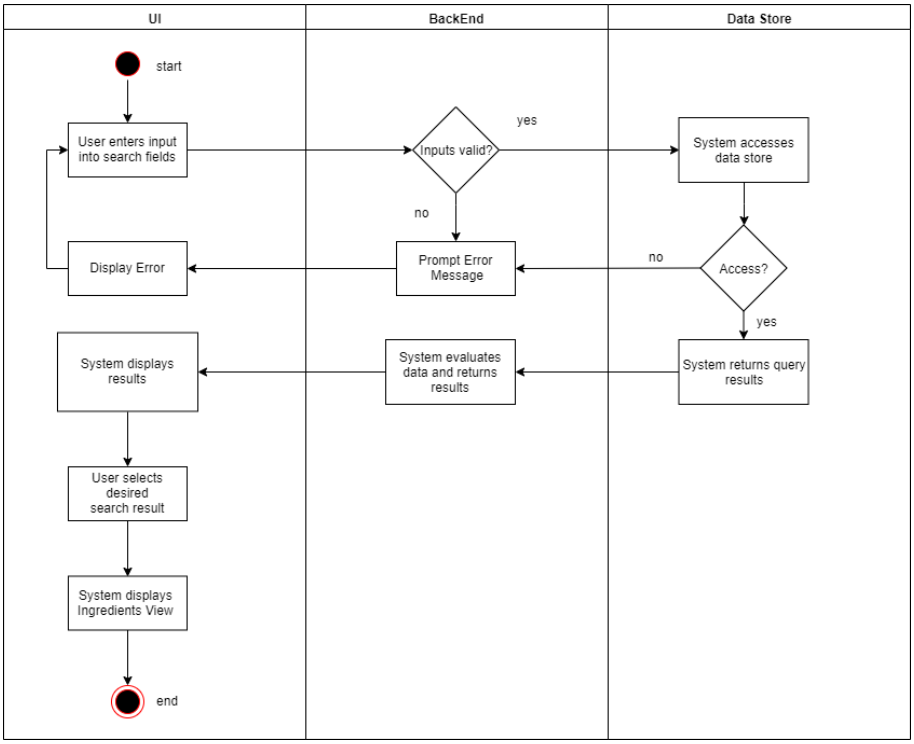
- Must be logged in.
- Must be at the Ingredient View.



Search For An Ingredient

Preconditions:

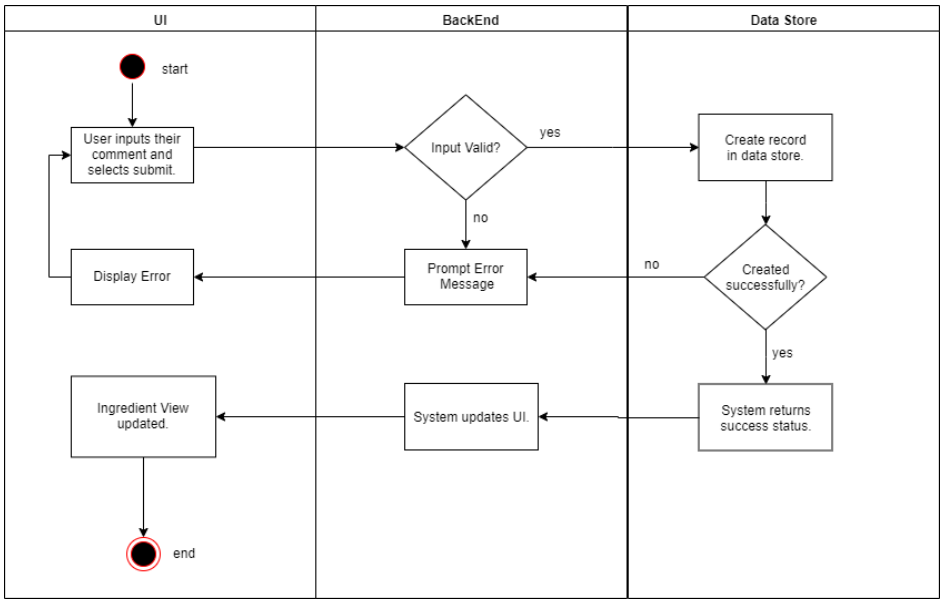
- Must be at the Home View.



Comment On An Upload

Preconditions:

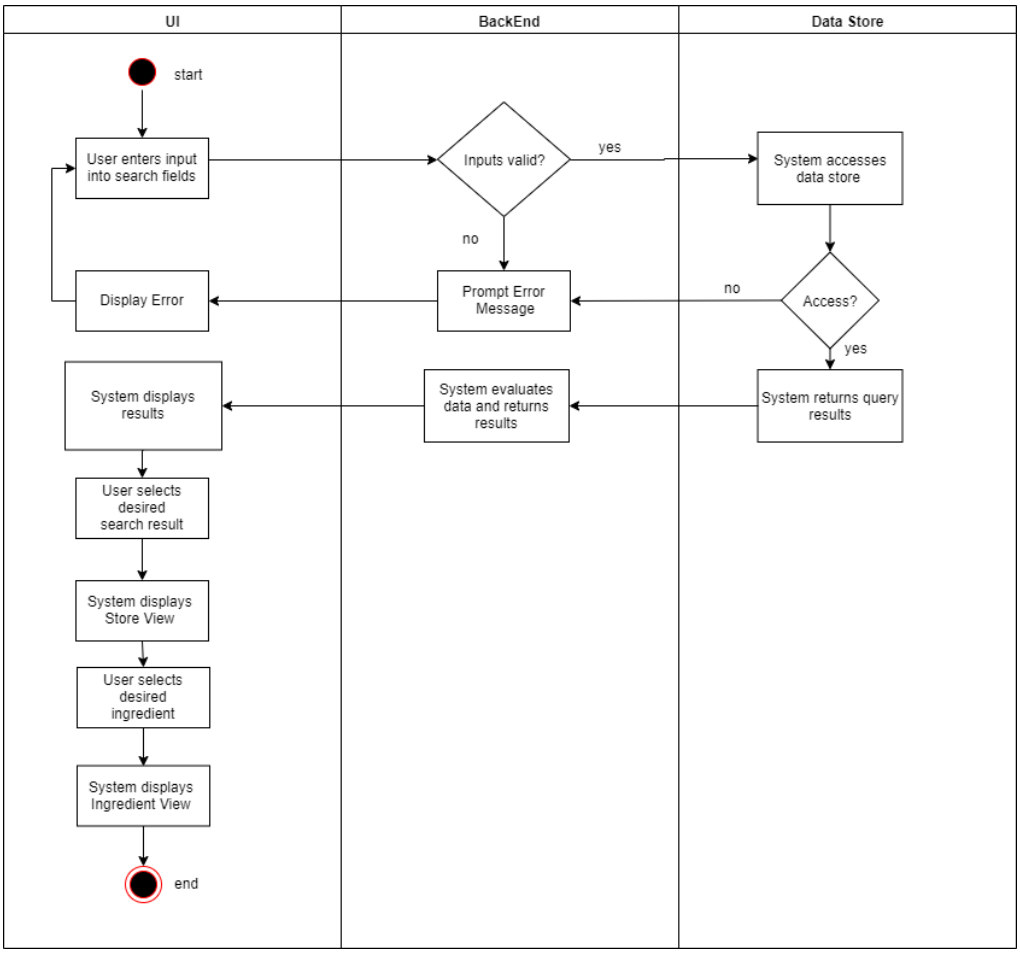
- User must be logged in.
- User must be at an Ingredient View.



Search For A Store

Preconditions:

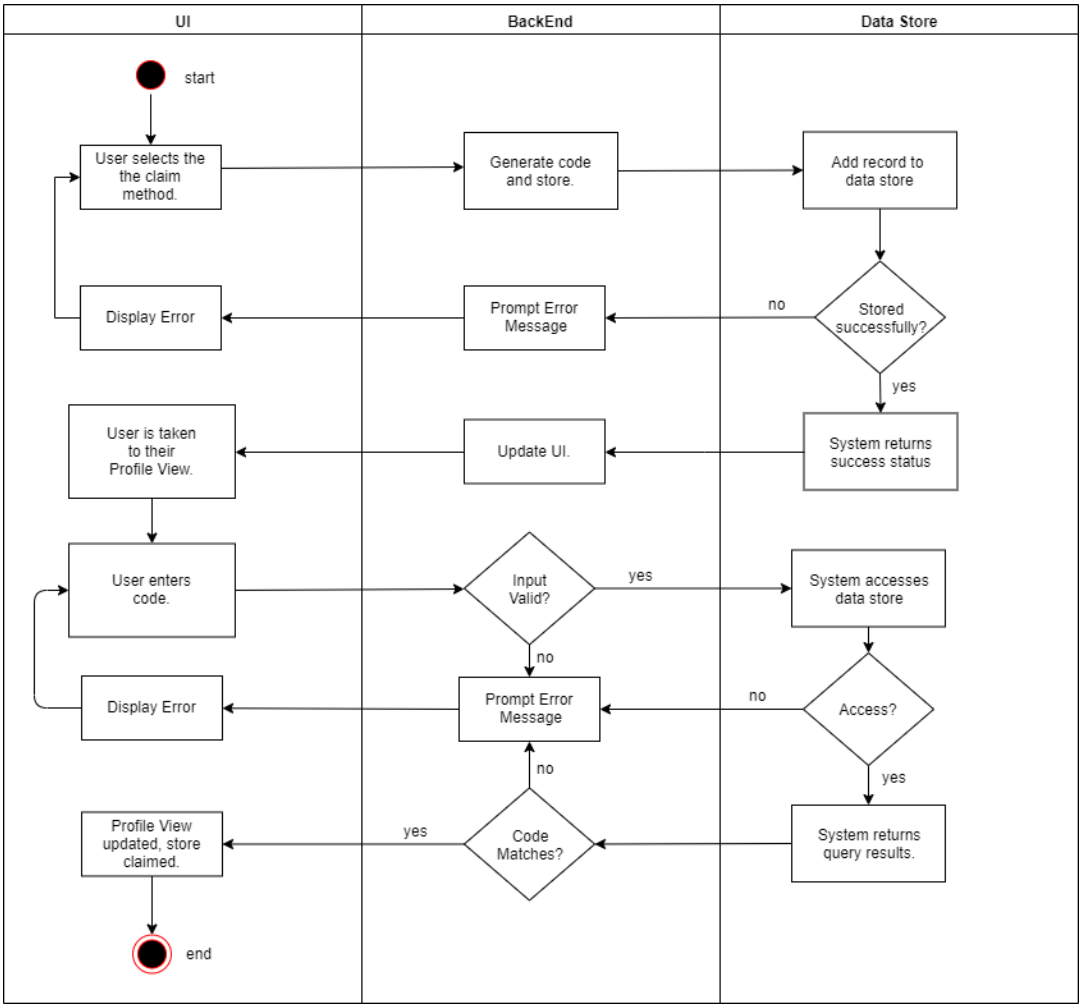
- User must be at the Home View.



Claim Store

Preconditions:

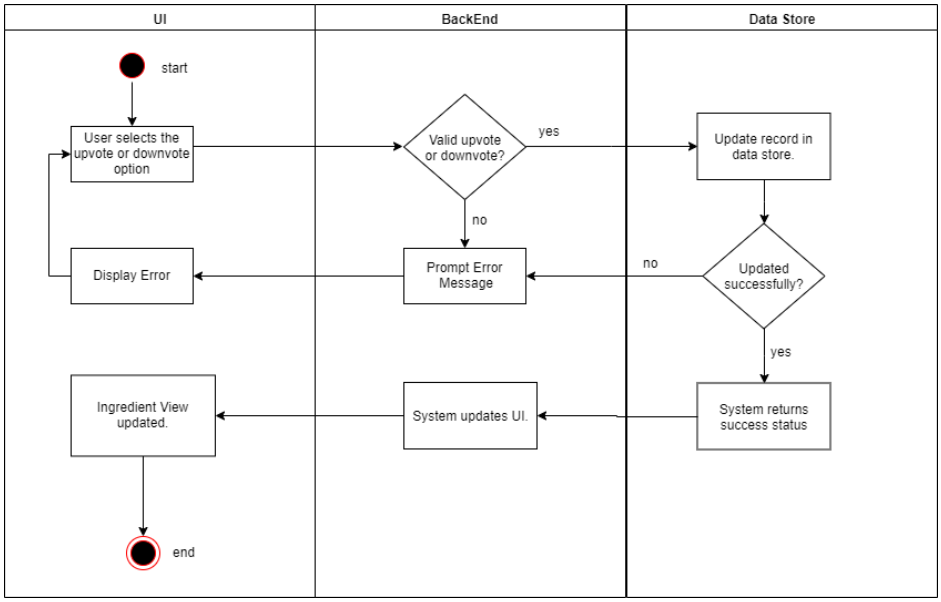
- User must be logged in.
- User must be on an unclaimed store's Store View.



Upvote / Downvote (Upload Or Store Owner's Description)

Preconditions:

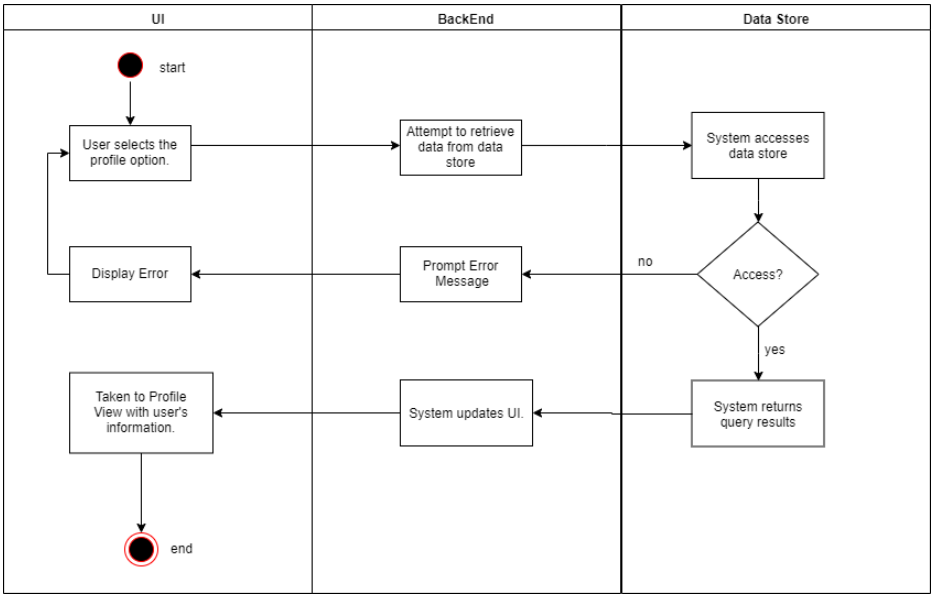
- Must be logged in.
- Must be at an Ingredient View.



Display Profile View

Preconditions:

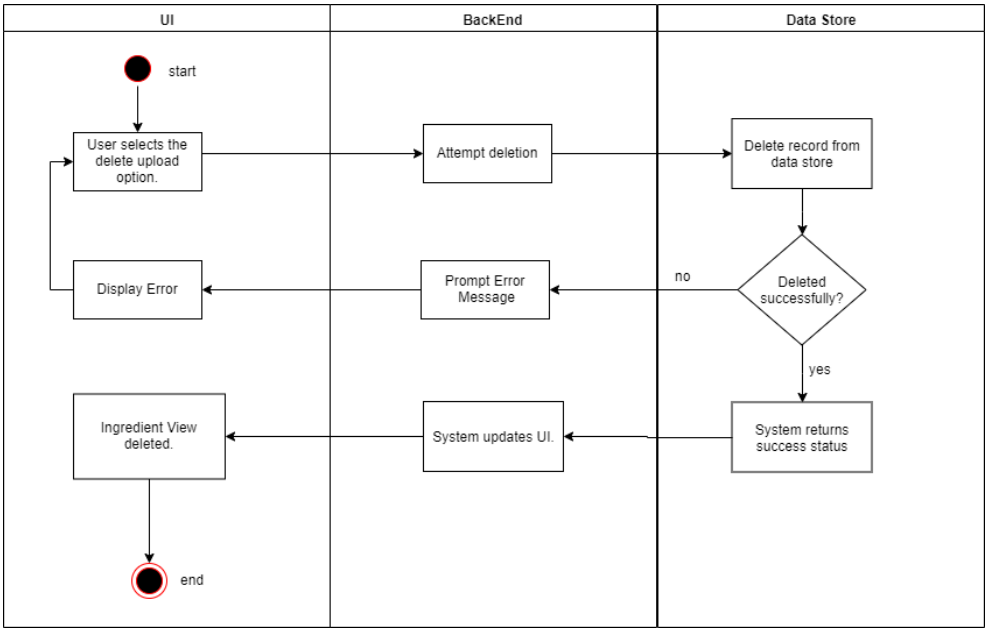
- User must be logged in.
- User must be at the Home View.



Admin Delete Upload

Preconditions:

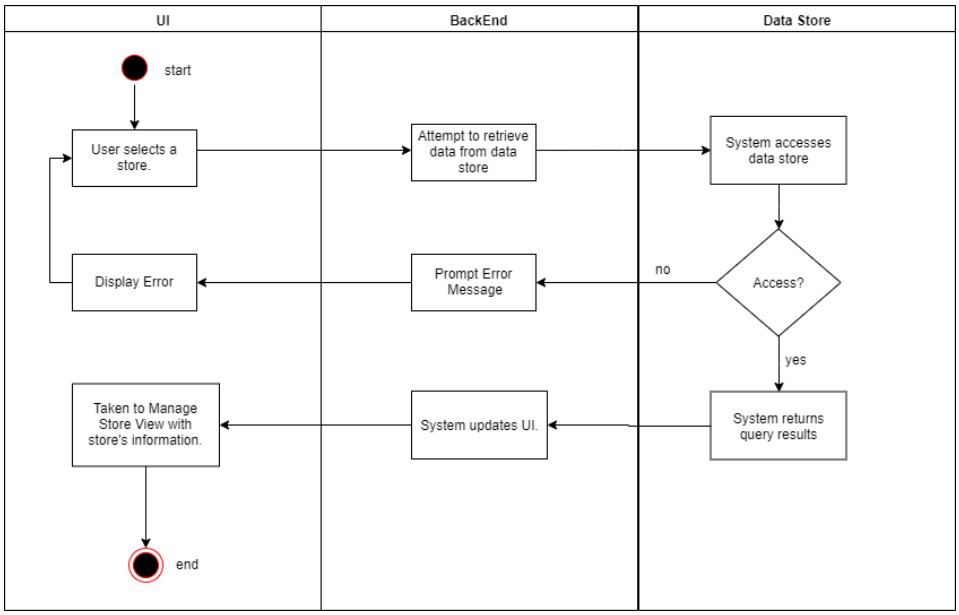
- User must be logged in with admin privileges.
- User must be at an Ingredient View.



Display Manage Store View.

Preconditions:

- User must be logged in and have Store Owner privileges.
- User must be at the Profile View.

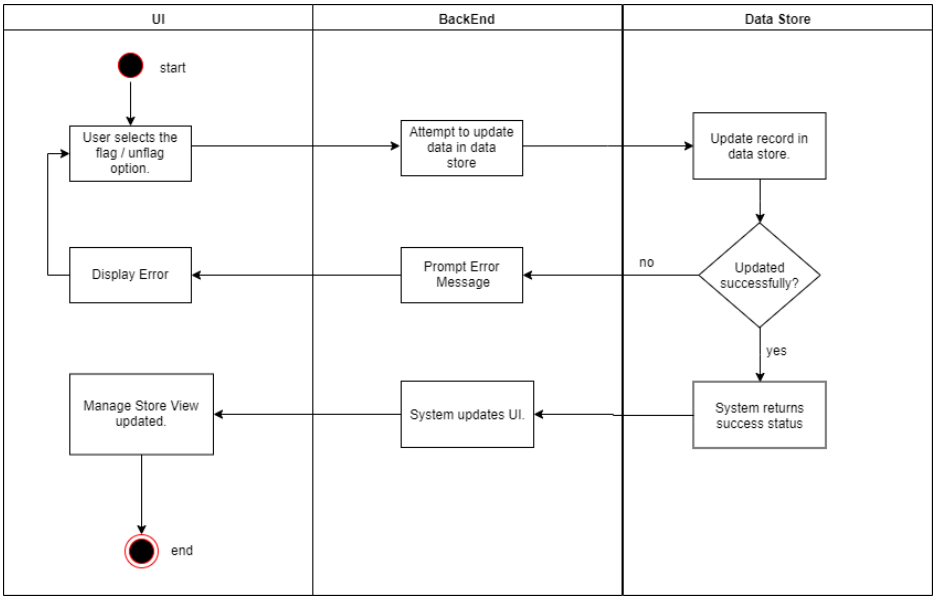


Manage Store - Flag / Unflag An Ingredient In Their Store

Preconditions:

- User must be Store Owner.

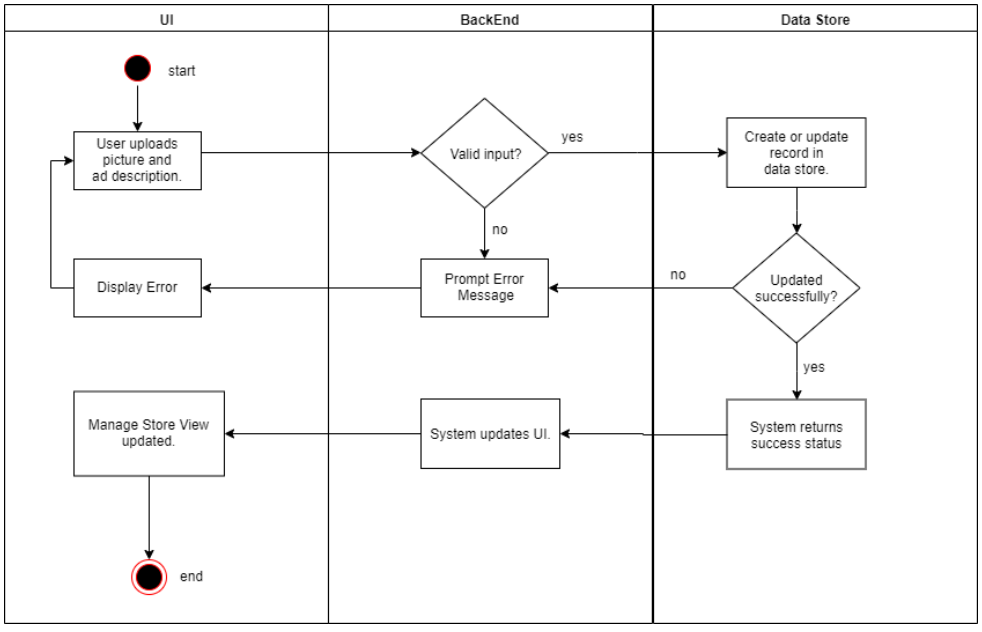
- User must be on Manage Store View.



Manage Store - Advertisement

Preconditions:

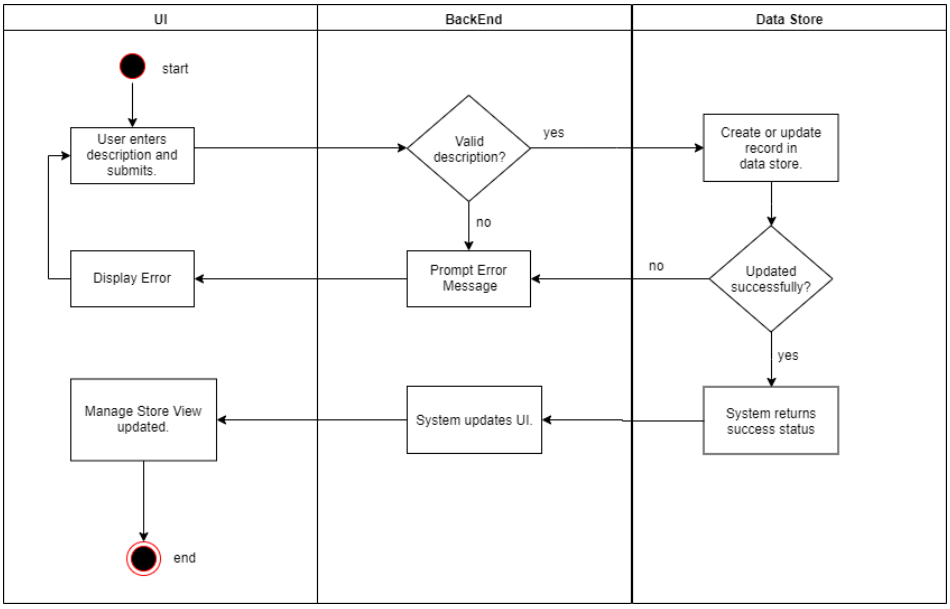
- User must be logged in and have Store Owner privileges.
- User must be at the Manage Store View.



Manage Store - Submit Description Of An Ingredient

Preconditions:

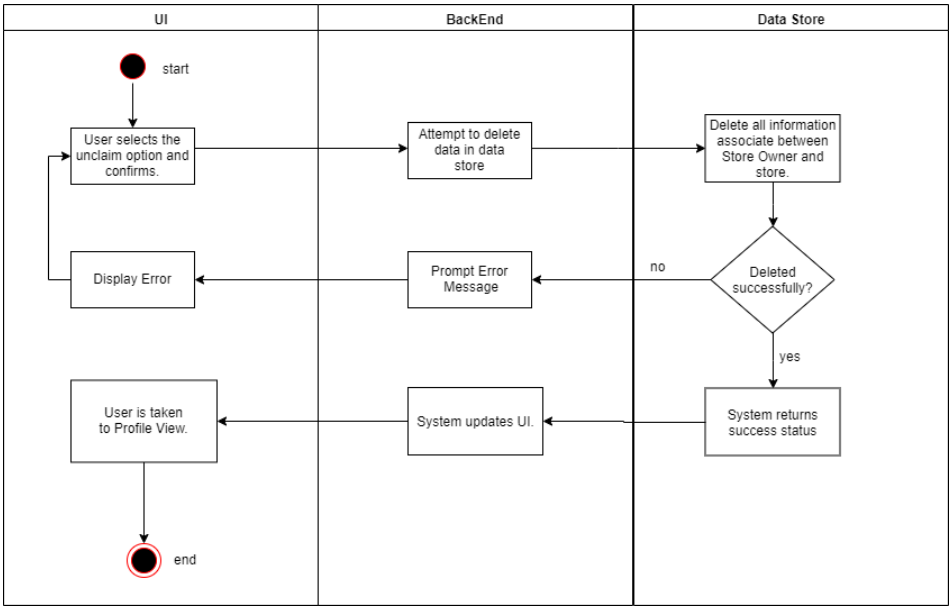
- User must be logged in with Store Owner privileges.
- User must be on the Manage Store View.



Manage Store - Unclaim Store

Preconditions:

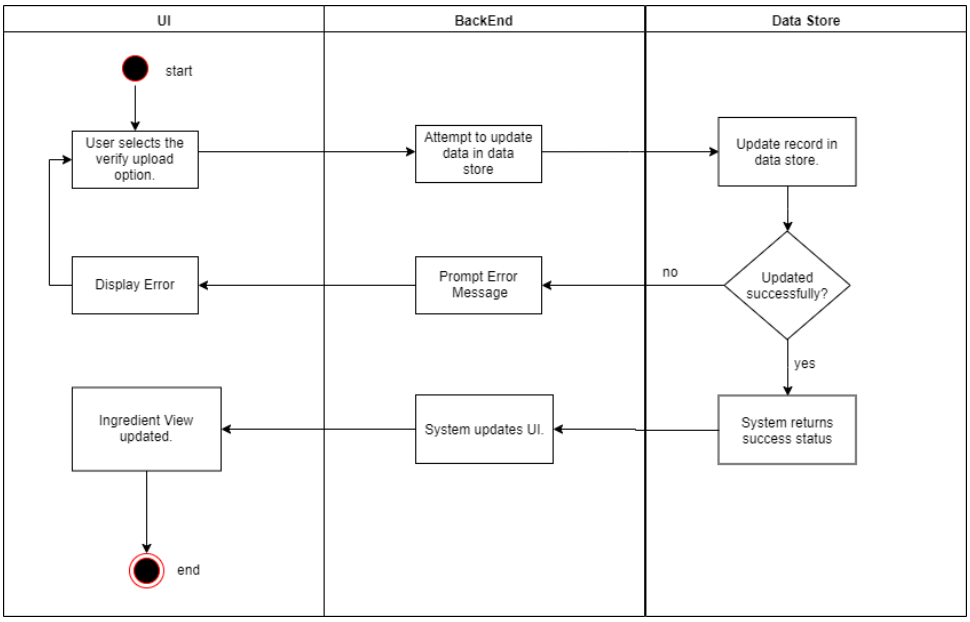
- User must be logged in and have Store Owner privileges.
- User must be on the Manage Store View.



Verify An Upload

Preconditions:

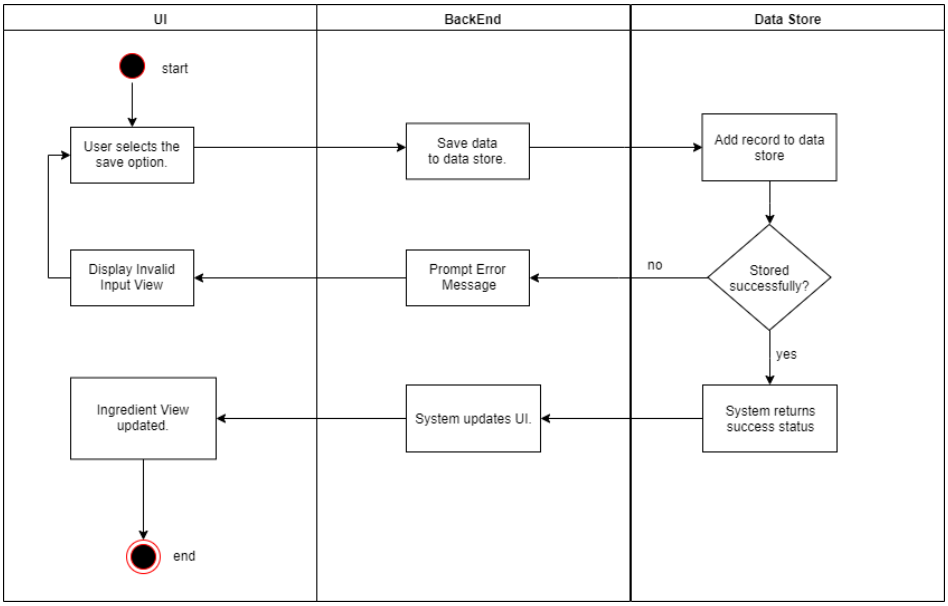
- User must be logged in and have Store Owner privileges.
- User must be on an Ingredient View for an ingredient from their store.



Ingredient Save List - Add To Save List

Preconditions:

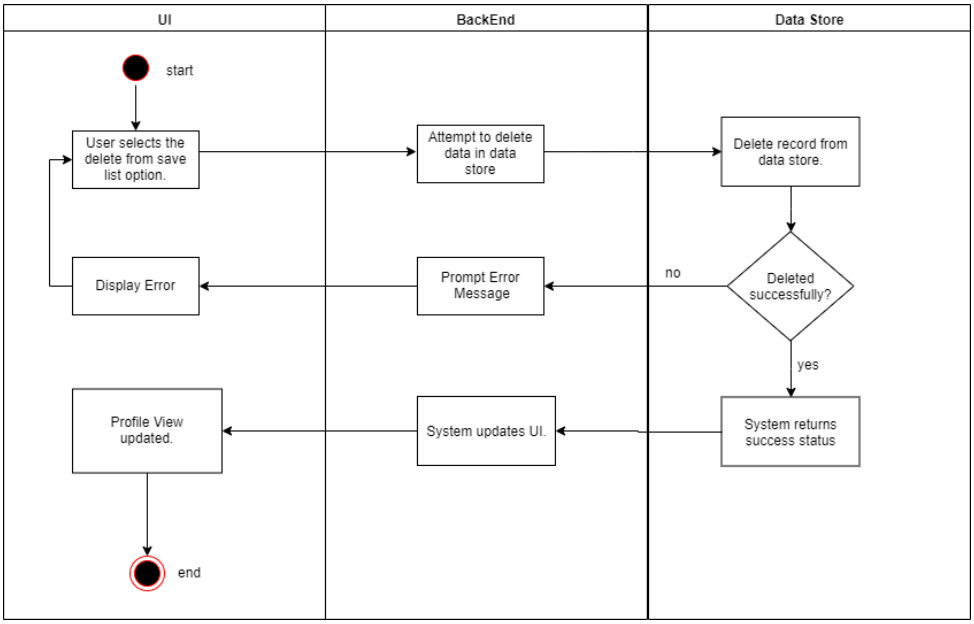
- User must be logged in.
- User must be at the Profile View.



Ingredient Save List - Delete From Save List

Preconditions:

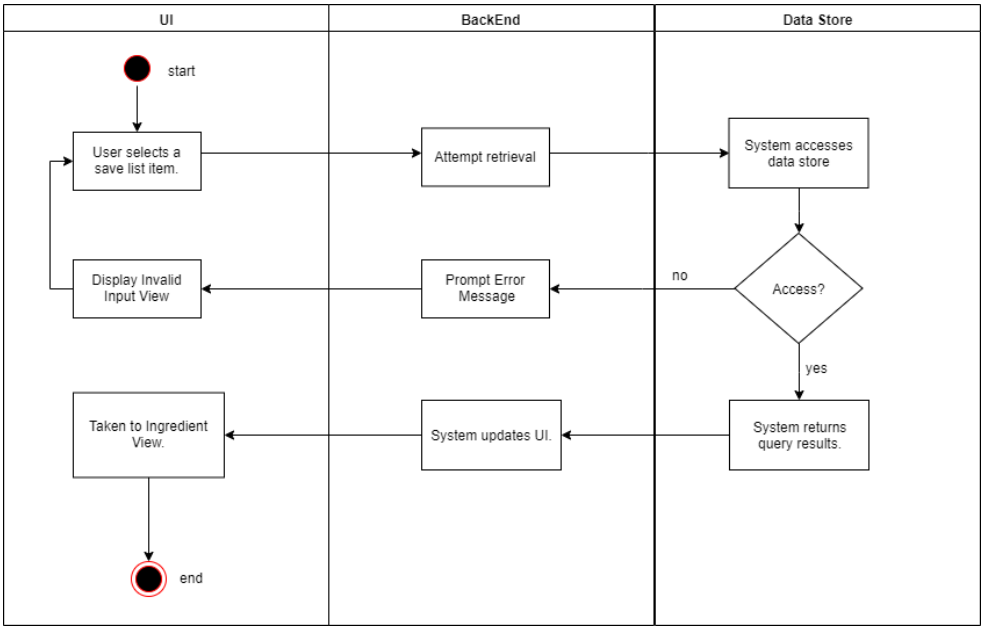
- User must be logged in.
- User must have a saved item on their save list.
- User must be at the Profile View.



Ingredient Save List - View Item From Save List

Preconditions:

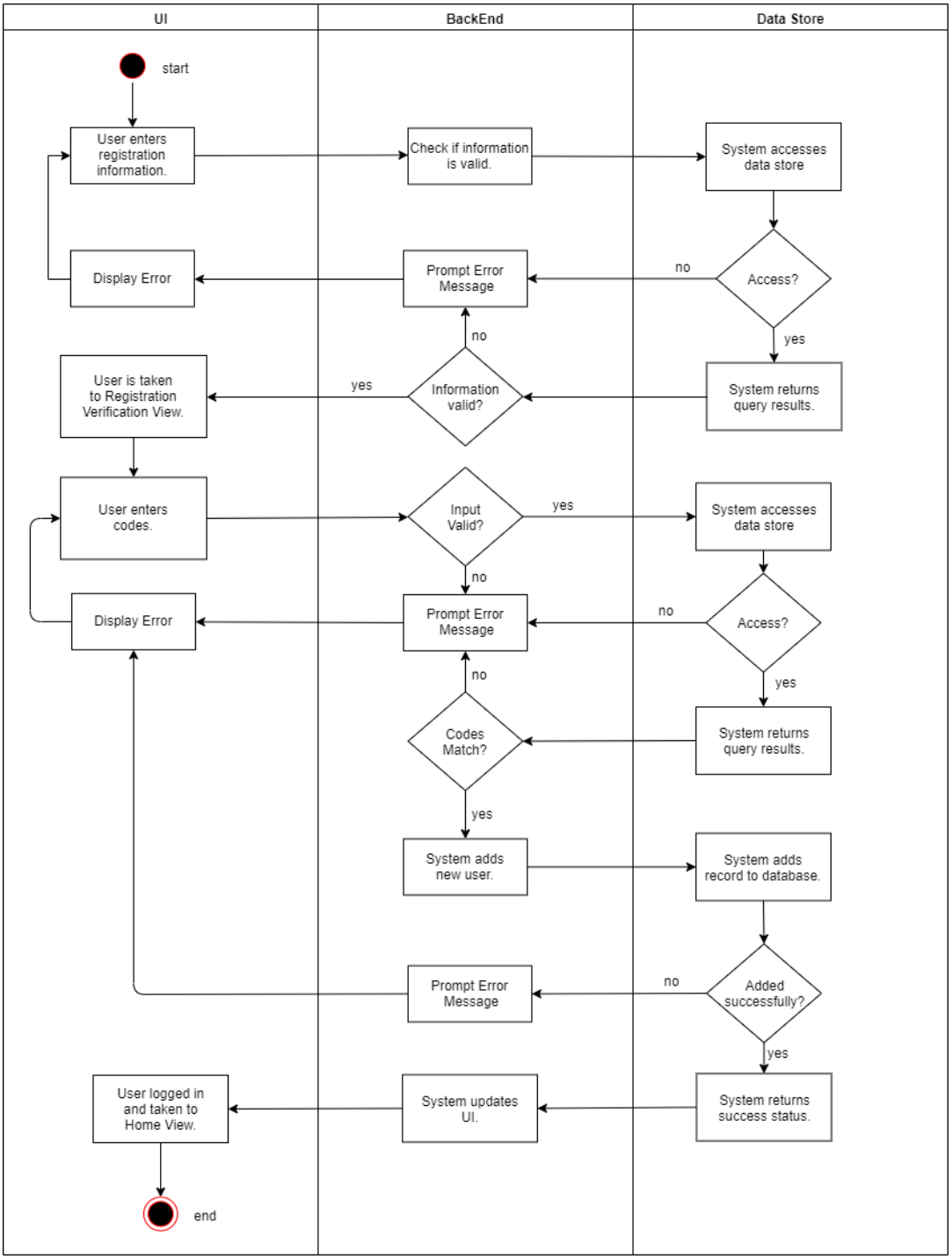
- User must be logged in.
- User must have a saved item on their save list.
- User must be at the Profile View.



Registration

Preconditions:

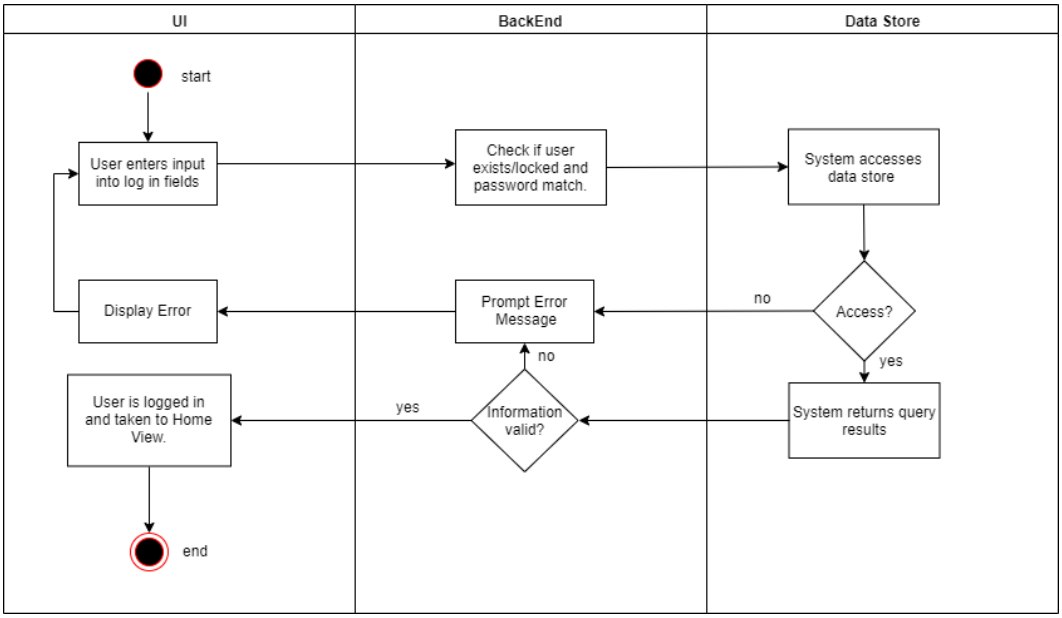
- User must not be logged in.



Authentication - Login

Preconditions:

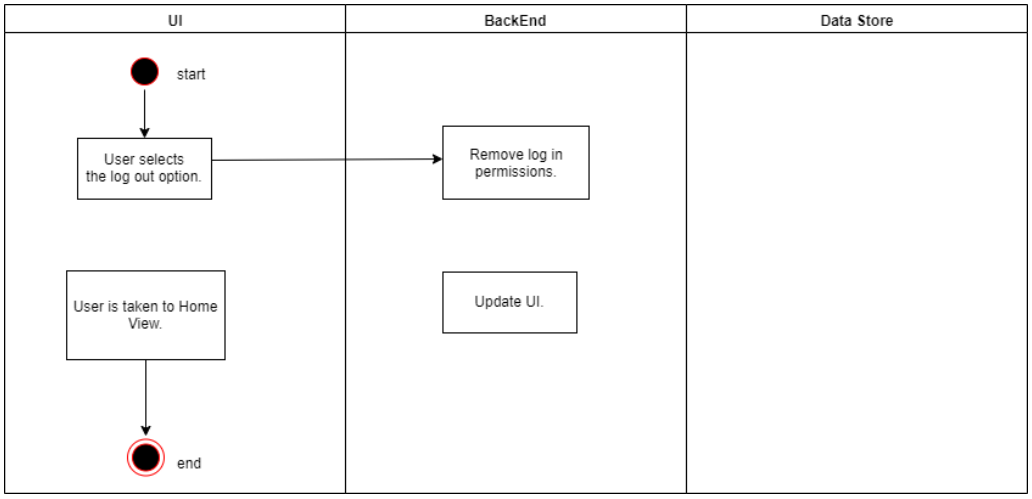
- User must be not logged in.
- User must have a registered account.



Authentication - Logout

Preconditions:

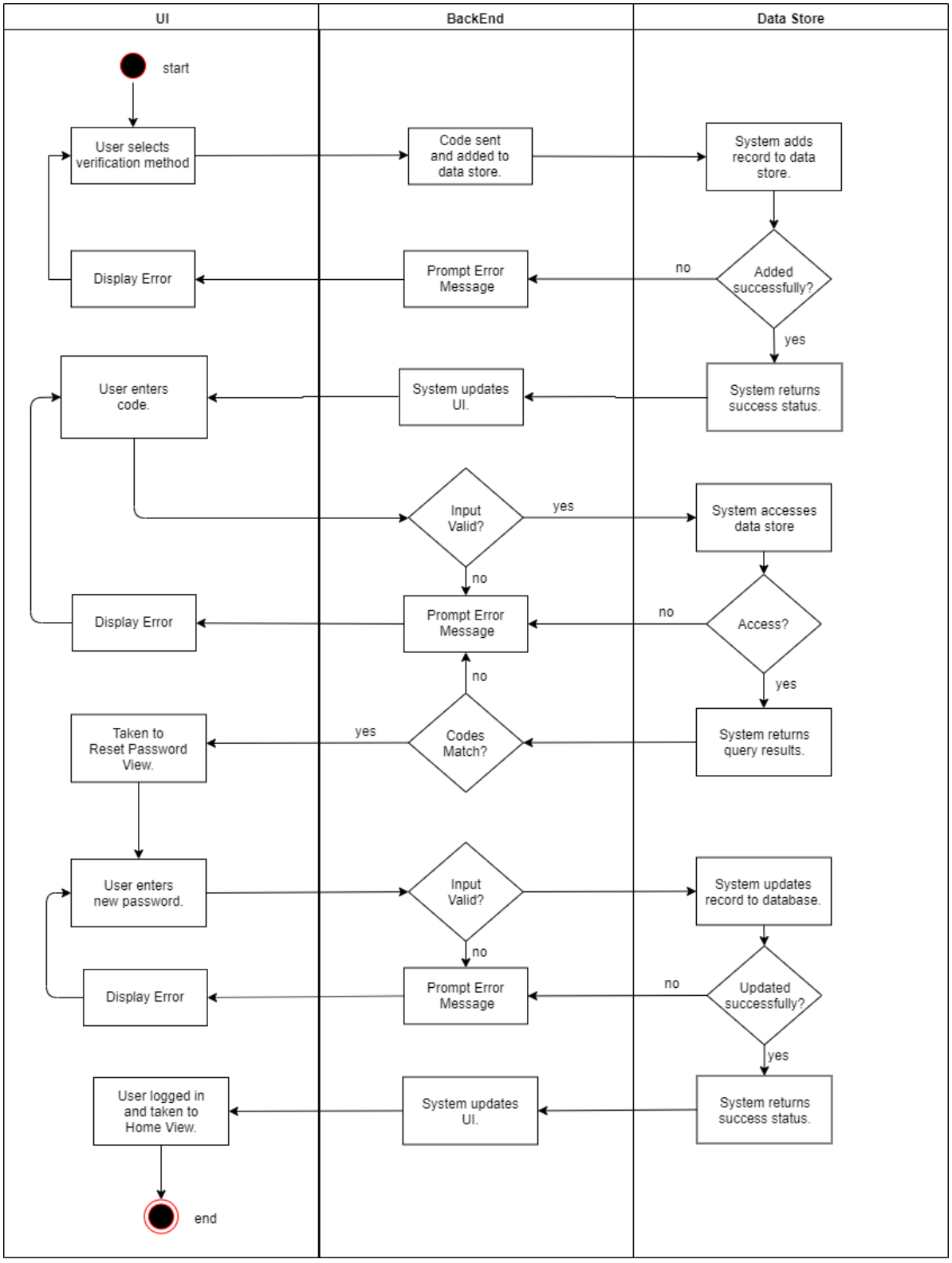
- User must be logged in.



Authentication - Reset Password

Preconditions:

- User must not be logged in to an account.
- User must be on Log In View.

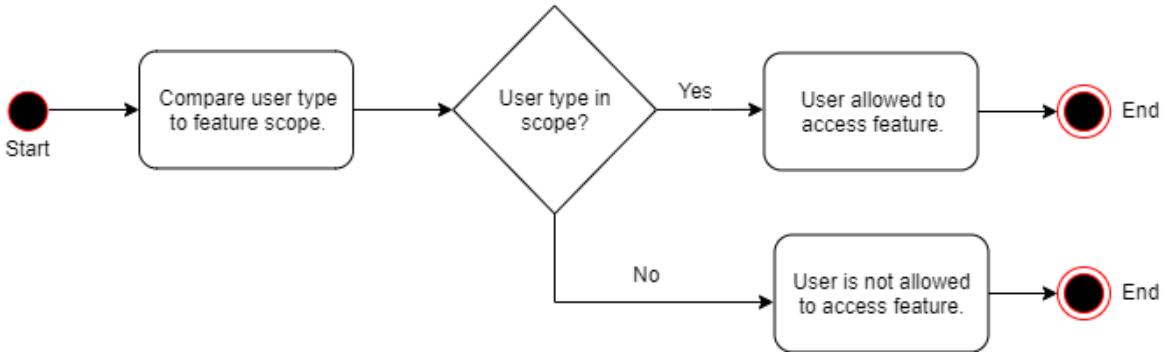

UI
BackEnd
Data Store
start
User selects verification method
Code sent and added to data store.
System adds record to data store.
Display Error
Prompt Error Message
no
Added successfully?
yes
User enters code.
System updates UI.
System returns success status.
Input Valid?
yes
System accesses data store
no
Display Error
Prompt Error Message
no
Access?
no
yes
Taken to Reset Password View.
yes
Codes Match?
System returns query results.
User enters new password.
Input Valid?
System updates record to database.
no
Display Error
Prompt Error Message
no
Updated successfully?
yes
User logged in and taken to Home View.
System updates UI.
System returns success status.
end

THE FOLLOWING DIAGRAMS DO NOT INCLUDE UI, SO THEY WILL BE REPRESENTED AS ACTIVITY DIAGRAMS

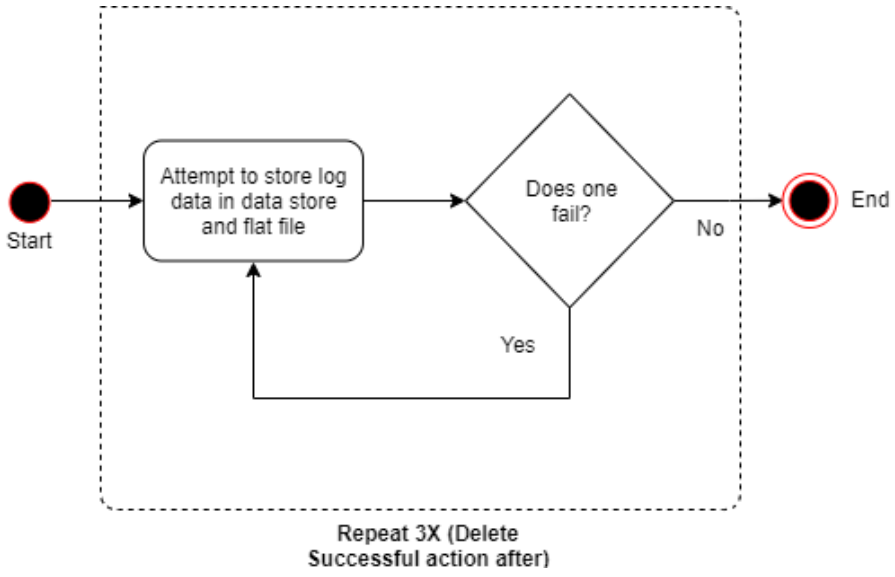
Authorization

Preconditions:

- User attempts to interact with a feature.



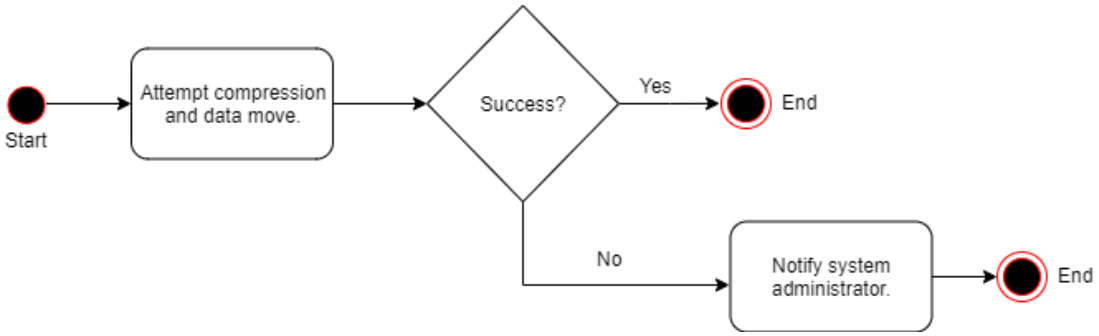
Logging



Archiving

Preconditions:

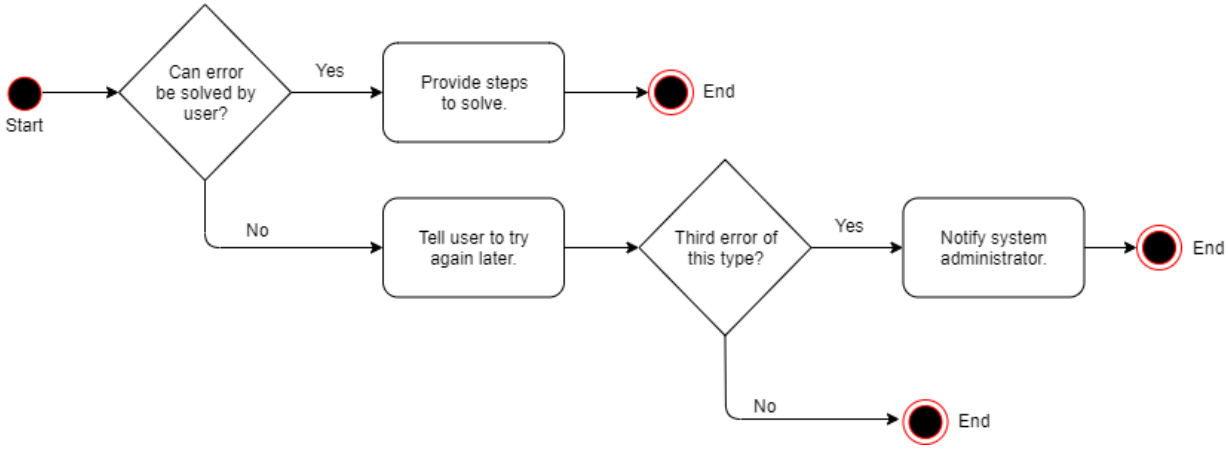
- Access to logs, 30 days or older.



Error Handling

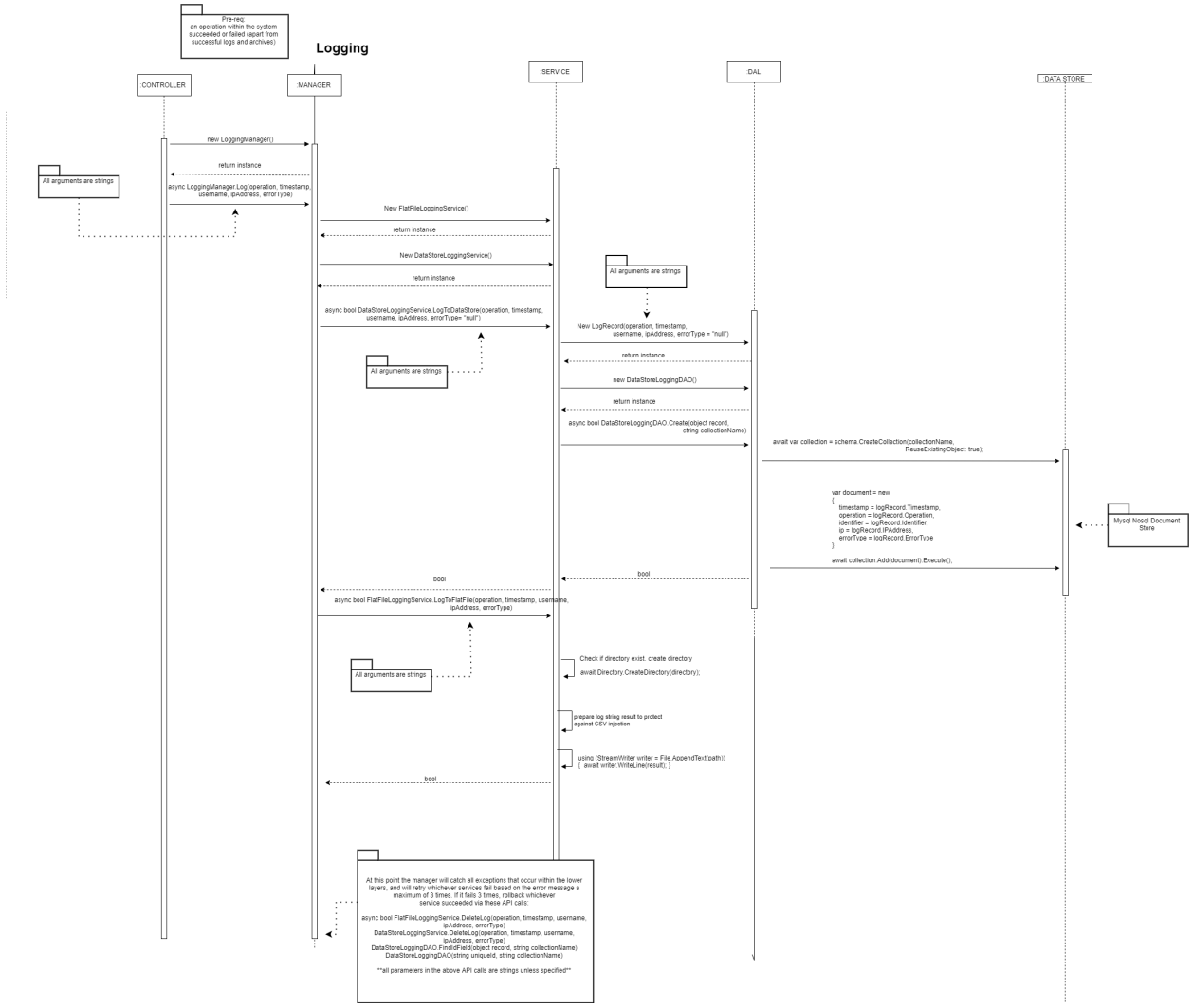
Preconditions:

- A system error occurs.

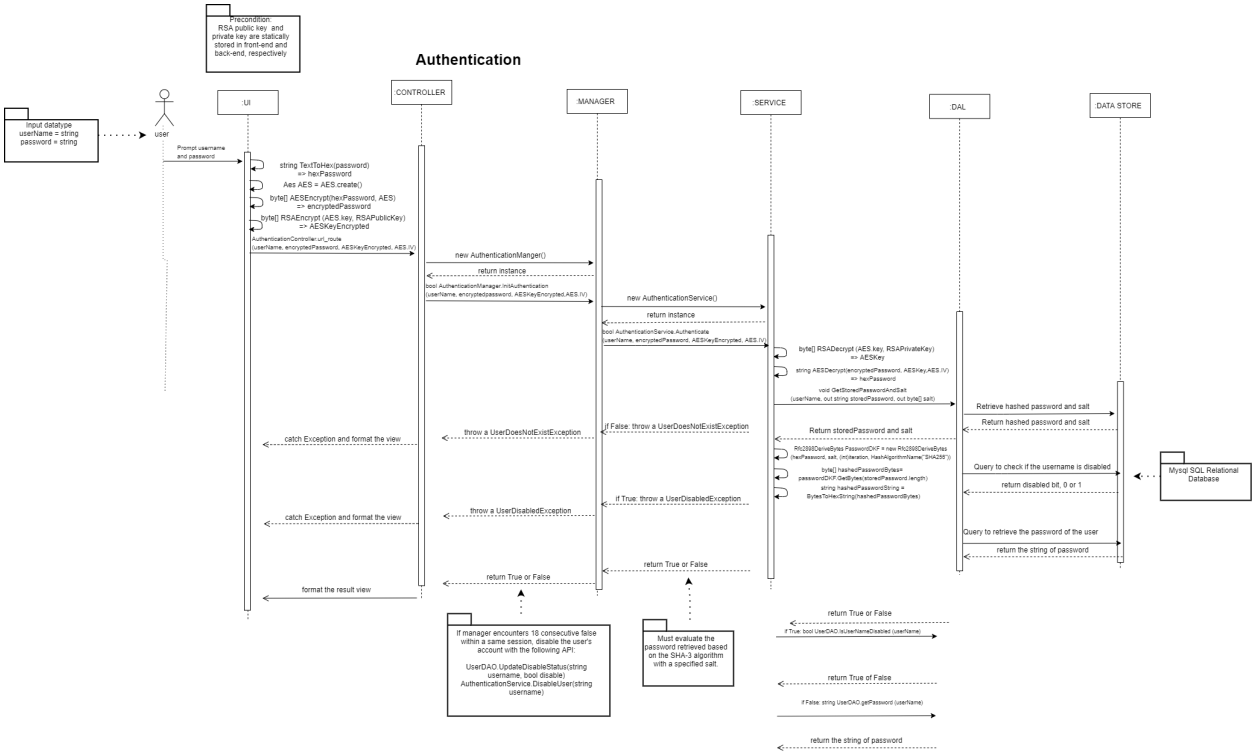


Low-Level Diagrams

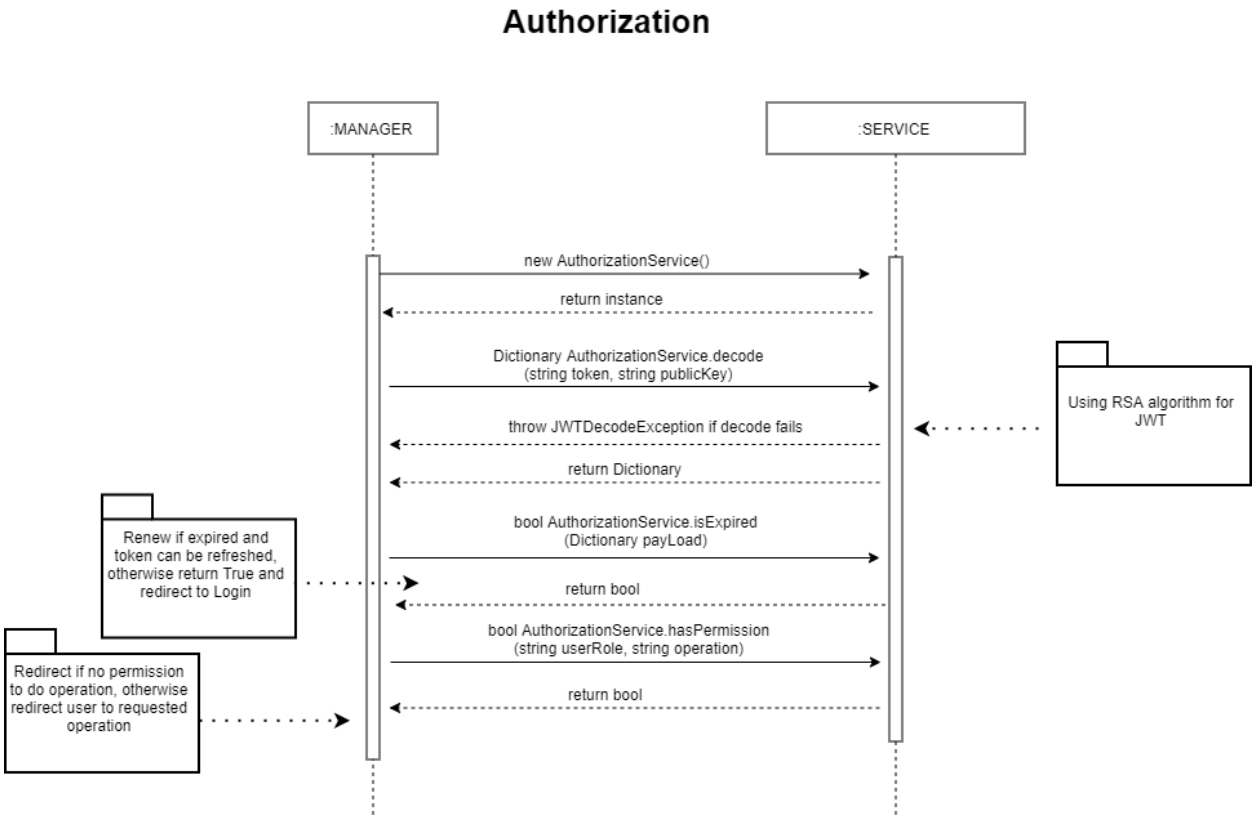
Logging Low Level



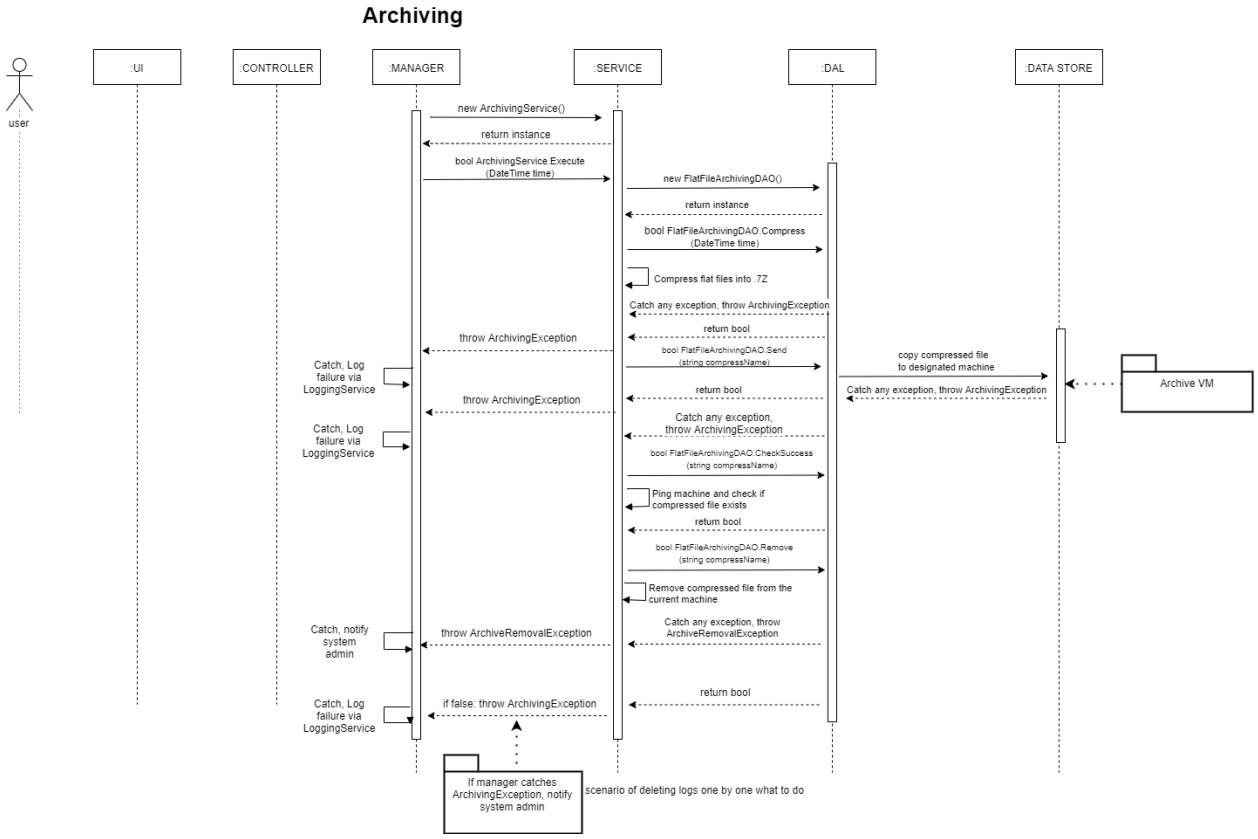
Authentication Low Level



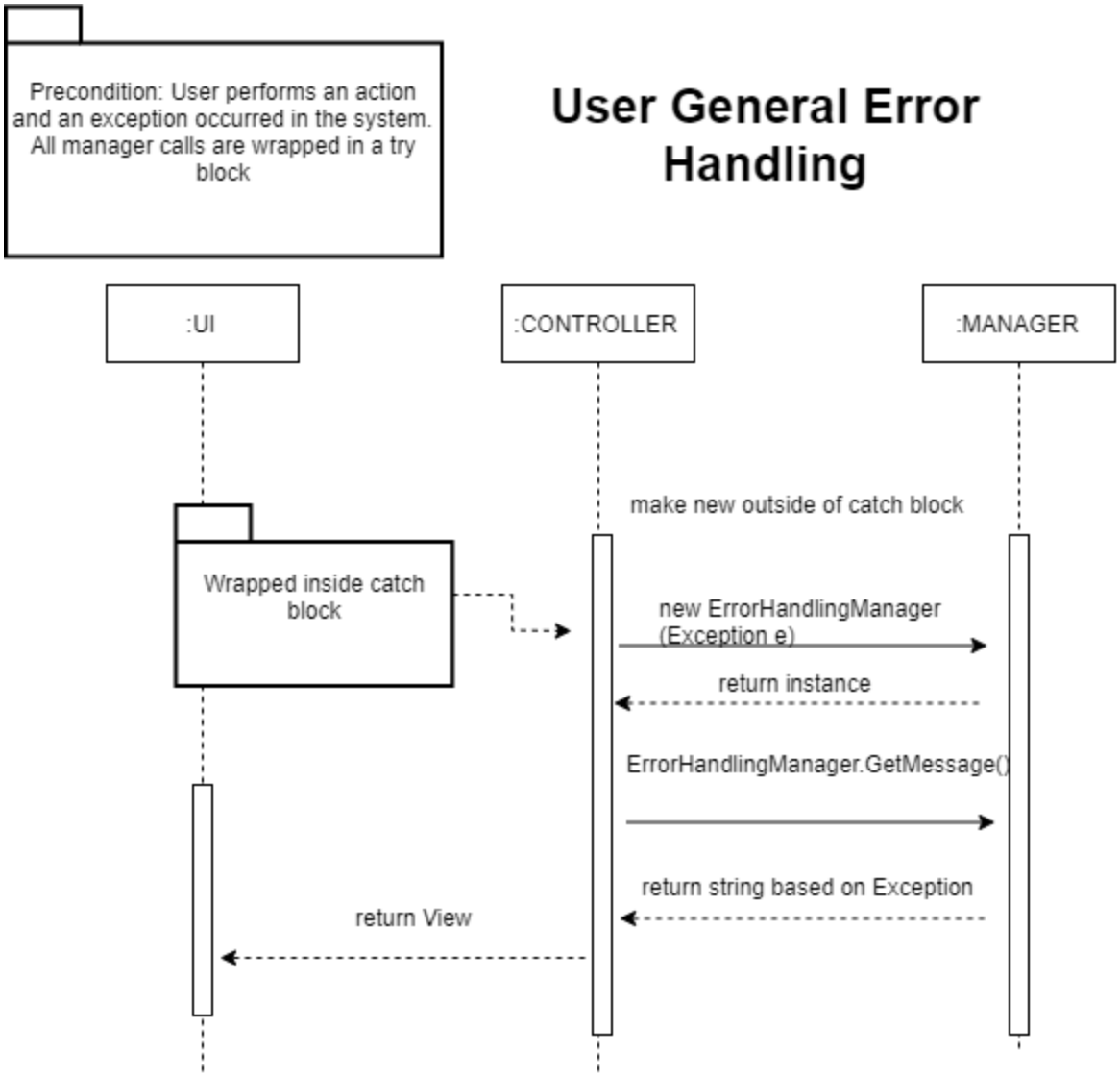
Authorization Low Level



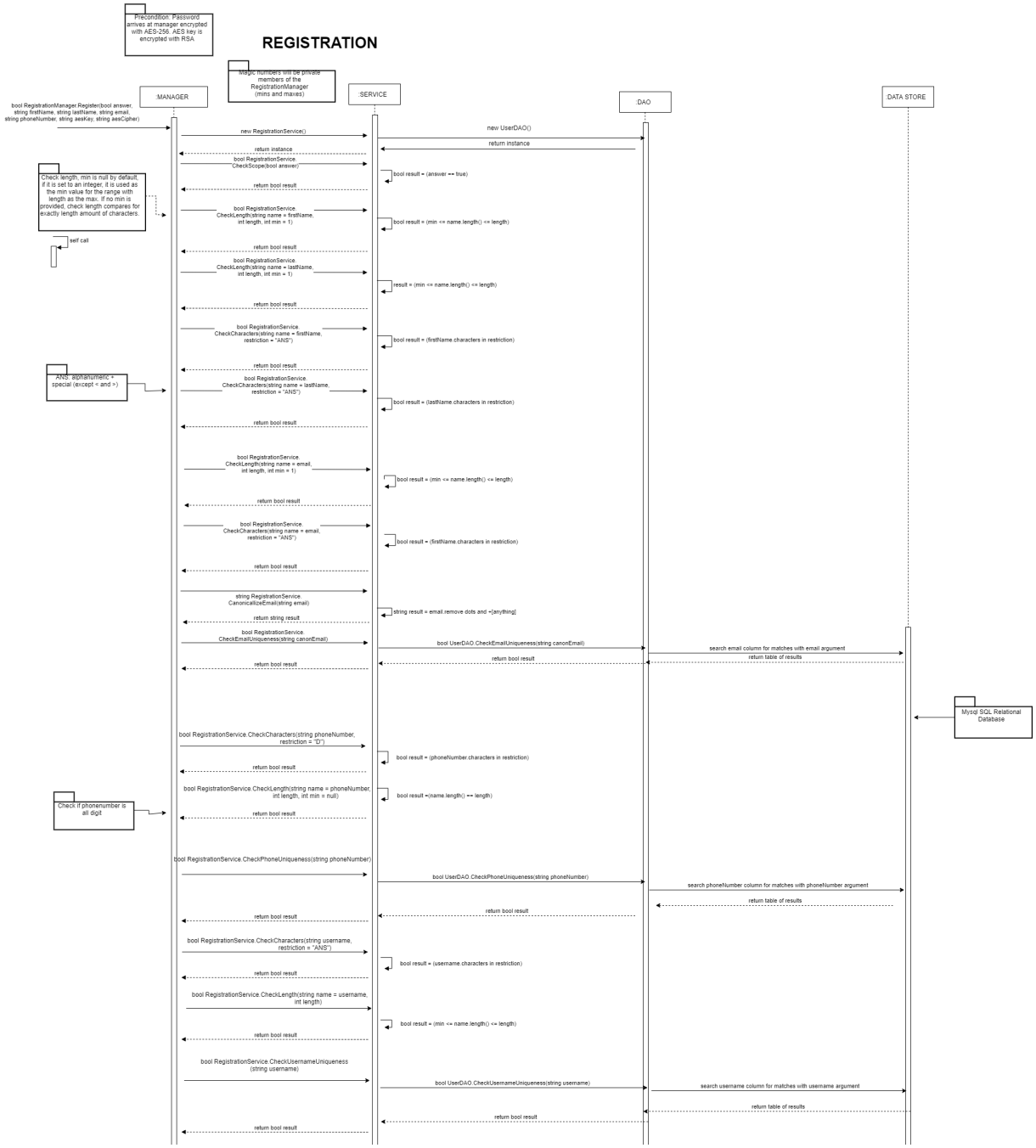
Archiving Low Level

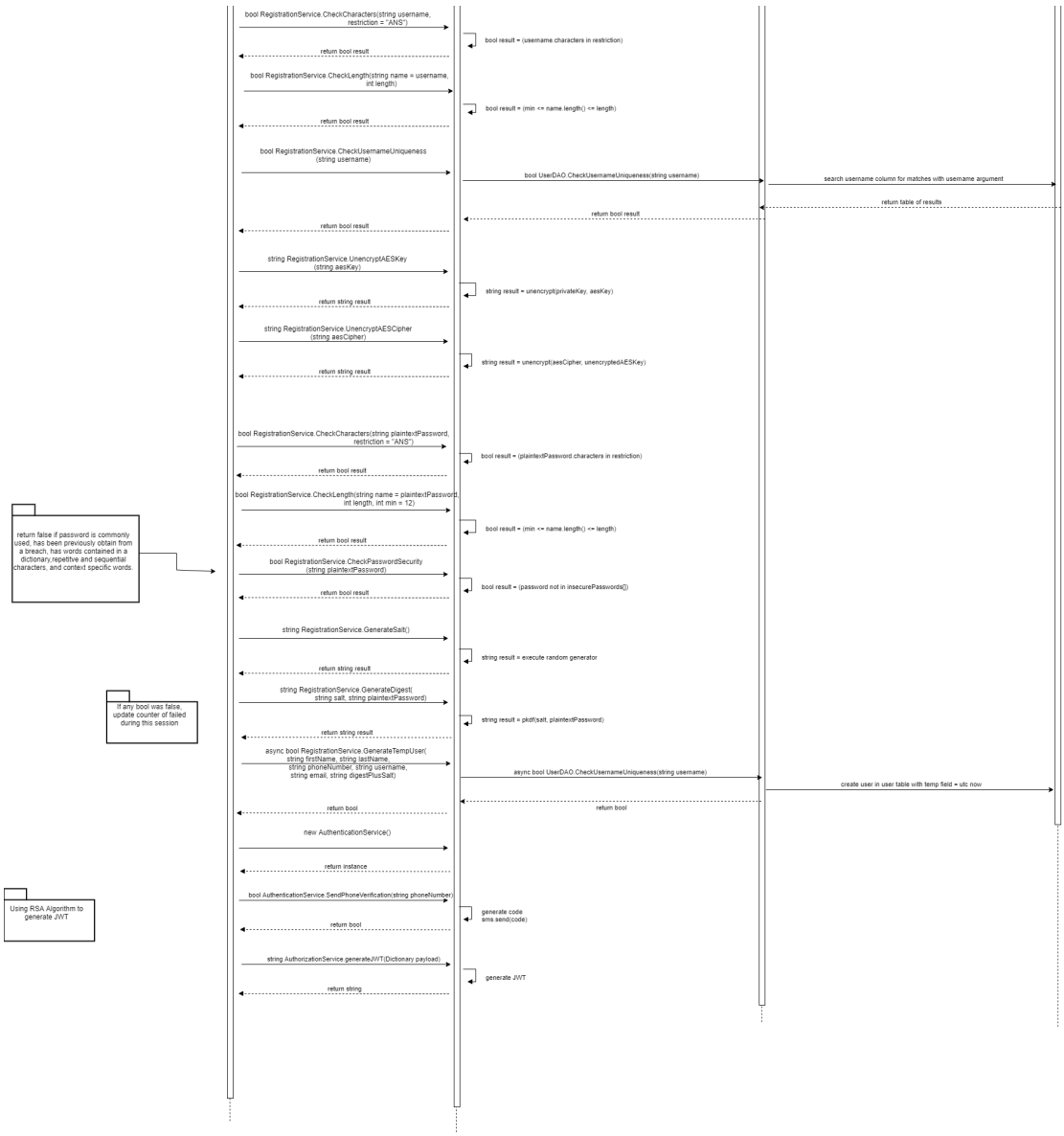


Error Handling Low Level

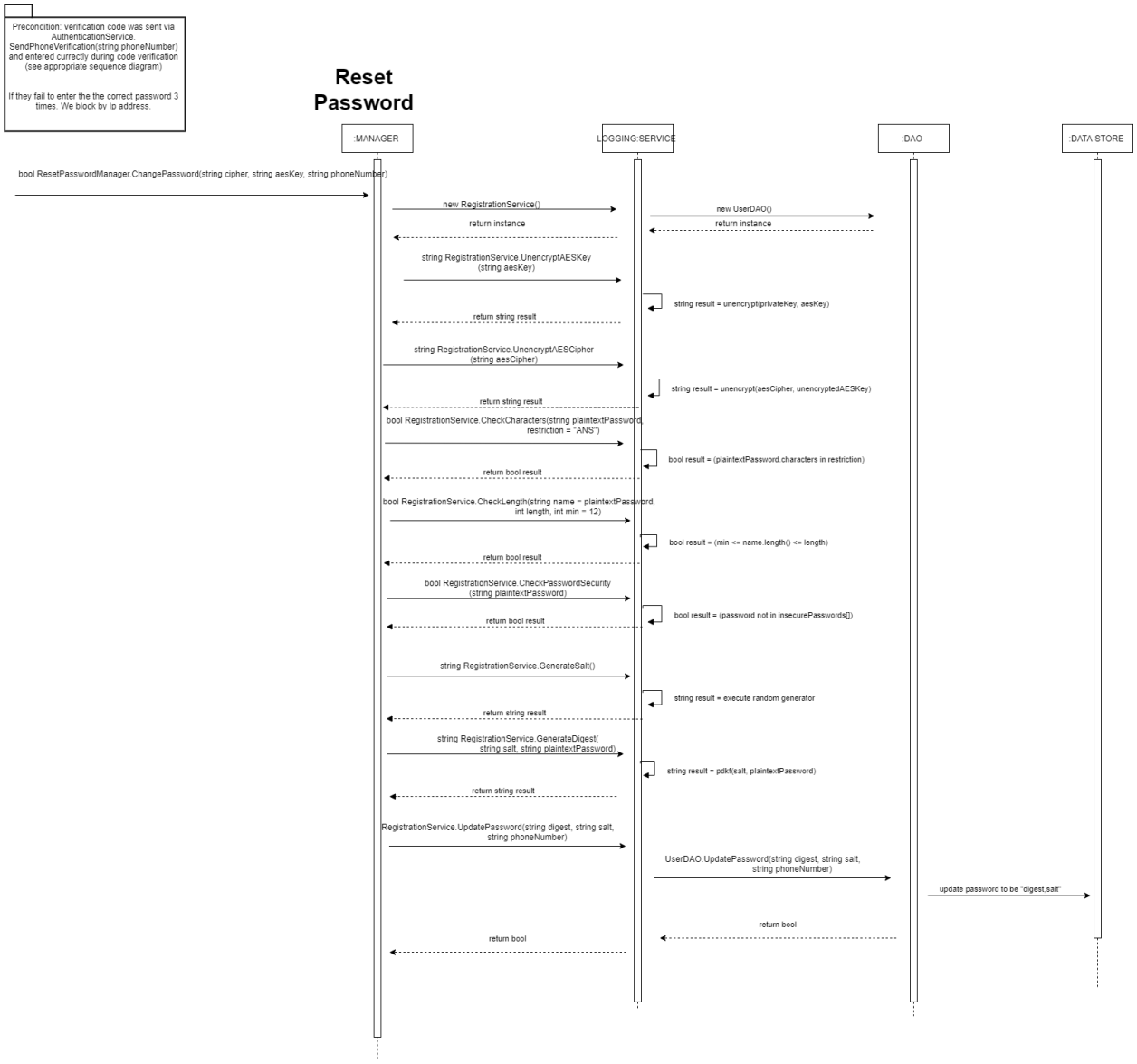
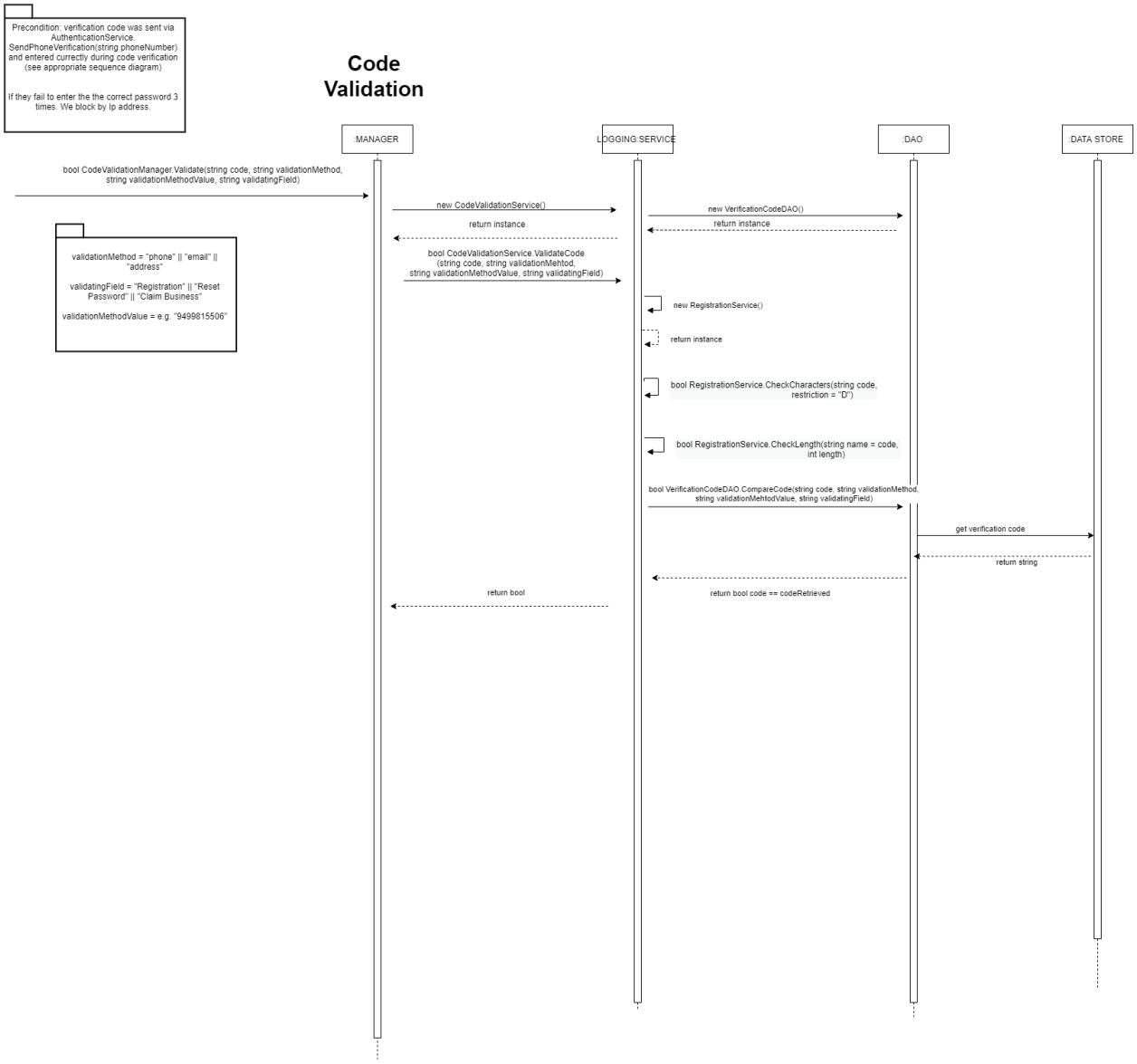


Registration Low Level




bool RegistrationService.CheckCharacters(string username, restriction = "ANS")
bool result = (username.characters in restriction)
return bool result
bool RegistrationService.CheckLength(string name = username, int length)
bool result = (min <= name.length() <= length)
return bool result
bool RegistrationService.CheckUsernameUniqueness (string username)
bool UserDAO.CheckUsernameUniqueness(string username)
search username column for matches with username argument
return table of results
return bool result
return bool result
string RegistrationService.UnencryptAESKey (string aesKey)
string result = unencrypt(privateKey, aesKey)
return string result
string RegistrationService.UnencryptAESCipher (string aesCipher)
string result = unencrypt(aesCipher, unencryptedAESKey)
return string result
bool RegistrationService.CheckCharacters(string plaintextPassword, restriction = "ANS")
bool result = (plaintextPassword.characters in restriction)
return bool result
bool RegistrationService.CheckLength(string name = plaintextPassword, int length, int min = 12)
bool result = (min <= name.length() <= length)
return bool result
return false if password is commonly used, has been previously obtain from a breach, has words contained in a dictionary,repetitive and sequential characters, and context specific words.
bool RegistrationService.CheckPasswordSecurity (string plaintextPassword)
bool result = (password not in insecurePasswords[])
return bool result
string RegistrationService.GenerateSalt()
string result = execute random generator
return string result
string RegistrationService.GenerateDigest(string salt, string plaintextPassword)
string result = pkdf(salt, plaintextPassword)
If any bool was false, update counter of failed during this session
return string result
async bool RegistrationService.GenerateTempUser(string firstName, string lastName, string phoneNumber, string username, string email, string digestPlusSalt)
async bool UserDAO.CheckUsernameUniqueness(string username)
create user in user table with temp field = utc now
return bool
return bool
new AuthenticationService()
return instance
Using RSA Algorithm to generate JWT
bool AuthenticationService.SendPhoneVerification(string phoneNumber)
generate code
sms.send(code)
return bool
string AuthorizationService.generateJWT(Dictionary payload)
generate JWT
return string

Reset Password Low Level



Code Validation Low Level