u

**Revision and Sign-Off Sheet**

**Change Record**

| Author | Version | Change reference | Date |
| --- | --- | --- | --- |
| Vo Dai Luong | 0.1 | Initialize | 10/10/2024 |
| Tran Tien Minh | 0.2 | Add some use case description | 12/10/2024 |
| Tran Tien Minh | 0.3 | Update the use case diagram, introduction, and use case description | 16/10/2024 |
| Tran Tien Minh | 0.4 | Update use case diagram, business rules, message list | 25/10/2024 |
| Vo Dai Luong | 1.0 | Upload list and view description | 02/11/2024 |

**Reviewers**

| Name | Company | Version | Position | Date |
| --- | --- | --- | --- | --- |
| Vo Dai Luong | UIT | 0.1 | Application Owner | 11/10/2024 |
| Vo Dai Luong | UIT | 0.2 | Application Owner | 13/10/2024 |
| Vo Dai Luong | UIT | 0.3 | Application Owner | 18/10/2024 |
| Vo Dai Luong | UIT | 0.4 | Application Owner | 28/10/2024 |
| Vo Dai Luong | UIT | 1.0 | Application Owner | 05/11/2024 |

**Table of Contents**

[**1. Introduction 6**](#_heading=h.gjdgxs)

[**1.1. Purpose 6**](#_heading=h.30j0zll)

[**1.2. Scope 6**](#_heading=h.1fob9te)

[**1.3. Intended Audiences and Document Organization 6**](#_heading=h.3znysh7)

[**1.4. References 6**](#_heading=h.3dy6vkm)

[2. Functional Requirements 7](#_heading=)

[**2.1. Use Case Description 7**](#_heading=h.4d34og8)

[UC1: Sign in 7](#_heading=h.2s8eyo1)

[Activities Flow 8](#_heading=)

[Business Rules 8](#_heading=)

[UC2: Sign Up 9](#_heading=h.rfhgx1ty7hpj)

[Activities Flow 10](#_heading=)

[Business Rules 10](#_heading=h.rwop98mgwn57)

[UC3: Forgot Password 12](#_heading=h.nt4pgyky7gtc)

[Activities Flow 14](#_heading=)

[Business Rules 15](#_heading=h.1q63jqdaa17e)

[UC4: Change Password 17](#_heading=h.70xd08jjrevt)

[Activities Flow 18](#_heading=h.w5z7rvt0z07n)

[Business Rules 18](#_heading=h.xppykhi2eg95)

[UC5: Ban an account 19](#_heading=h.xd763kl0qhqg)

[Activities Flow 20](#_heading=h.mg86x6jad02t)

[Business Rules 20](#_heading=h.gr087yrcdu0e)

[UC6:Unban an account 24](#_heading=h.7vcc1dw1was2)

[Activities Flow 24](#_heading=h.qxr0493f608w)

[Business Rules 24](#_heading=h.7qczahxjgqoz)

[UC7: Filter/Search post 26](#_heading=h.mug4ba1d52ws)

[Activities Flow 27](#_heading=h.9a4oxlvcmdis)

[Business Rules 28](#_heading=h.57slzktx1gzv)

[UC8: Save Post to Favorites 28](#_heading=h.s1etdlrkqly8)

[Activities Flow 29](#_heading=h.u5nnybp9jlon)

[Business Rules 29](#_heading=h.gn33ke64polf)

[UC9: Suggest Similar Posts 30](#_heading=h.sod99k7icqwx)

[Activities Flow 31](#_heading=h.gfslejvdh93p)

[Business Rules 31](#_heading=h.2yov9w8kueyl)

[UC10: View Post 32](#_heading=h.oc4wc3f6vq3t)

[Activities Flow 33](#_heading=h.f1uaqy6gd1tp)

[Business Rules 33](#_heading=h.awc18ocnnr6x)

[UC11: Create Post 34](#_heading=h.oap4f75c266o)

[Activities Flow 35](#_heading=h.z98spq9g711o)

[Business Rules 35](#_heading=h.4i1r5fwaugpo)

[UC12: Update Post 36](#_heading=h.kkj4d6o5c7gk)

[Activities Flow 37](#_heading=h.vv83jqjcxiw9)

[Business Rules 38](#_heading=h.7cwzn6m7f7sn)

[UC13: Delete Post 38](#_heading=h.advkii60xryf)

[Activities Flow 39](#_heading=h.jfwhk12a0hqd)

[Business Rules 40](#_heading=h.hbg06k5ouk3b)

[UC14: Approve a Post 40](#_heading=h.rr5numkf9vet)

[Activities Flow 41](#_heading=h.4fjedktdyfok)

[Business Rules 41](#_heading=h.w0l812dxk2cq)

[UC15: Reject a Post 42](#_heading=h.qu6w47nkgp95)

[Activities Flow 43](#_heading=h.4wp4ljguoebo)

[Business Rules 43](#_heading=h.1zmaoolywwpq)

[UC16: Report a Post 44](#_heading=h.9hq74fxidgkn)

[Activities Flow 45](#_heading=h.dtahvxpm7bcl)

[Business Rules 45](#_heading=h.mh131df3a73g)

[UC17: Leave Feedback 46](#_heading=h.yholoi8bk93x)

[Activities Flow 47](#_heading=h.cbrt0yfys8of)

[Business Rules 48](#_heading=h.1qiln2wd1yll)

[UC18: Link Sales Wallet 48](#_heading=h.e80309307j8p)

[Activities Flow 49](#_heading=h.mu8hqyaz05ek)

[Business Rules 50](#_heading=h.yg3s4e44wnpk)

[UC19: Rate Seller 51](#_heading=h.2lafvcawnbjr)

[Activities Flow 51](#_heading=h.dtbpcsapctq4)

[Business Rules 52](#_heading=h.7n7n7vs0n08l)

[UC20: Chat with Seller 52](#_heading=h.tyjhk56ptwub)

[Activities Flow 53](#_heading=h.q5as1csx9h4p)

[Business Rules 54](#_heading=h.c3m8ls8sgrod)

[UC21: Create a Category 54](#_heading=h.65n1vvuvk1bq)

[Activities Flow 55](#_heading=h.k5bnqsg45cg)

[Business Rules 55](#_heading=h.arcrp0lgv28t)

[UC22: Update a Category 56](#_heading=h.6tusvvnxhcpl)

[Activities Flow 57](#_heading=h.lx5dpumxbb7c)

[Business Rules 57](#_heading=h.a31136msan52)

[UC23: Delete a Category 58](#_heading=h.srlvdwwo1q02)

[Activities Flow 59](#_heading=h.qt2ucpcfgkwi)

[Business Rules 59](#_heading=h.zde9pmnq5l7x)

[UC24: Update information for the help center 60](#_heading=h.cj7dlh8xn5h6)

[Activities Flow 61](#_heading=h.oqy6w5o12qzc)

[Business Rules 61](#_heading=h.9n3amvbjb6kc)

[UC25: Handle feedback from the user 62](#_heading=h.ot70h85zqxpl)

[Activities Flow 63](#_heading=h.abgwf07clquz)

[Business Rule 63](#_heading=h.5zcryy12ifzq)

[**2.2. List Description 64**](#_heading=h.26in1rg)

[**2.3. View Description 64**](#_heading=h.35nkun2)

[**3. Non-functional Requirements 64**](#_heading=h.44sinio)

[User Access and Security 64](#_heading=h.2jxsxqh)

[Performance Requirements 65](#_heading=h.4i7ojhp)

[Implementation Requirements 66](#_heading=h.2xcytpi)

[**4. System Requirements 66**](#_heading=h.qsh70q)

[SharePoint Web Parts 66](#_heading=h.23ckvvd)

[Custom Pages 67](#_heading=h.ihv636)

[Scheduled Agents 67](#_heading=h.32hioqz)

[Technical Concern 67](#_heading=h.1hmsyys)

[**5. Appendixes 68**](#_heading=h.41mghml)

[Glossary 68](#_heading=h.2grqrue)

[Messages 68](#_heading=h.3fwokq0)

# Introduction

# Purpose

This document is the complete Software Requirements Specification (SRS) and Design Document for the MBX project, detailing the requirements and design guidelines for the development process. It is a key reference for developers, offering a roadmap for application functionality, task assignments, and deployment strategies.

This document's main objective is to define the software requirements for the MBX project and provide a structured design framework. It acts as a foundational guide for developers, project managers, and other stakeholders, ensuring a shared understanding of project goals, functionalities, and design principles.

# Scope

This document encompasses both the functional and non-functional requirements of the MBX project. It defines how the applications under development will operate, outlining features, constraints, and interfaces. The scope covers various aspects, including user interactions, system performance, security, and deployment considerations.

# Intended Audiences and Document Organization

This comprehensive document outlines the roles and responsibilities of various teams involved in the MBX project. The project encompasses the development, documentation, and user acceptance testing (UAT) of the application. Each team plays a crucial role in ensuring the project’s success, and this document aims to provide a detailed overview of their responsibilities.

This document is intended for:

* Development team: Responsible for developing detailed design, implementing and performing unit tests, integration tests, and system tests for the migrated application
* Data Migration team: Responsible for creating data migration scripts, and performing data migration for the application.
* Documentation Team: Responsible for writing a User Guide for the application.
* UAT team: Responsible for conducting user acceptance test sessions with end users.

Below are the main sections of the document:

* **1. Introduction**: This section describes the general introduction of this document.
* **2. Functional Requirements**: This section describesthe functional requirements in detail.
* **3. Non-functional Requirements:** This section describes the non-functional requirements of this application such as user access and security, interfaces, screens, and performance.
* **4. Other Requirements:** This section describes other requirements such as archive or security audit functions.
* **5.** **SharePoint Application Design:** This section describes the design of the SharePoint application.
* **6. Appendixes**: This section describes other requirements for this application and other supporting information for this document**.**
* NOTE:Please refer to section 6.1 for all acronyms and abbreviations in this document.

# References

| **#** | **Title** | **Version** | **File Name / Link** | **Description** |
| --- | --- | --- | --- | --- |
| 1 |  |  |  |  |
| 2 |  |  |  |  |

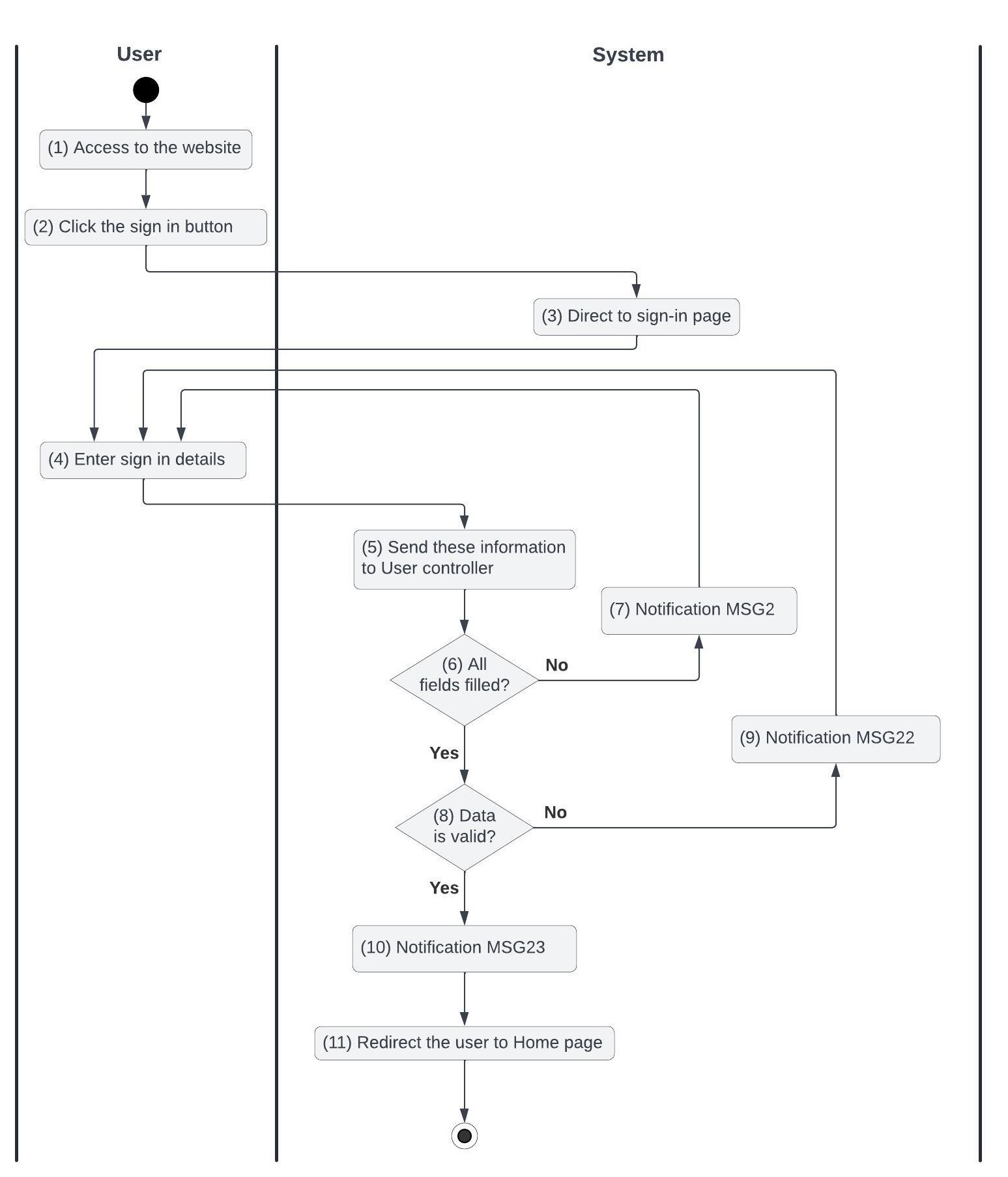
# Functional Requirements

# Use Case Description

### UC1: Sign in

| **Name** | **Sign In** |
| --- | --- |
| **Description** | This use case describes the process by which a user logs into the system |
| **Actor** | User |
| **Trigger** | * When a user clicks the ‘Sign in’ button |
| **Pre-condition** | * The system should be online and accessible * The user is not logged into the system * The sign-in button is clickable and actionable and then redirects the user to the sign-in page * The user has internet access and valid credentials to sign in * The user is on the sign-in page (refer to the ‘Sign In Form’ in the ‘View Description’ file) |
| **Post-condition** | * The user is successfully logged into the system * The user is redirected to the home page * A valid session or token is created for the user |

#### Activities Flow

**

*Figure 1: Activities Flow of the sign-in use case*

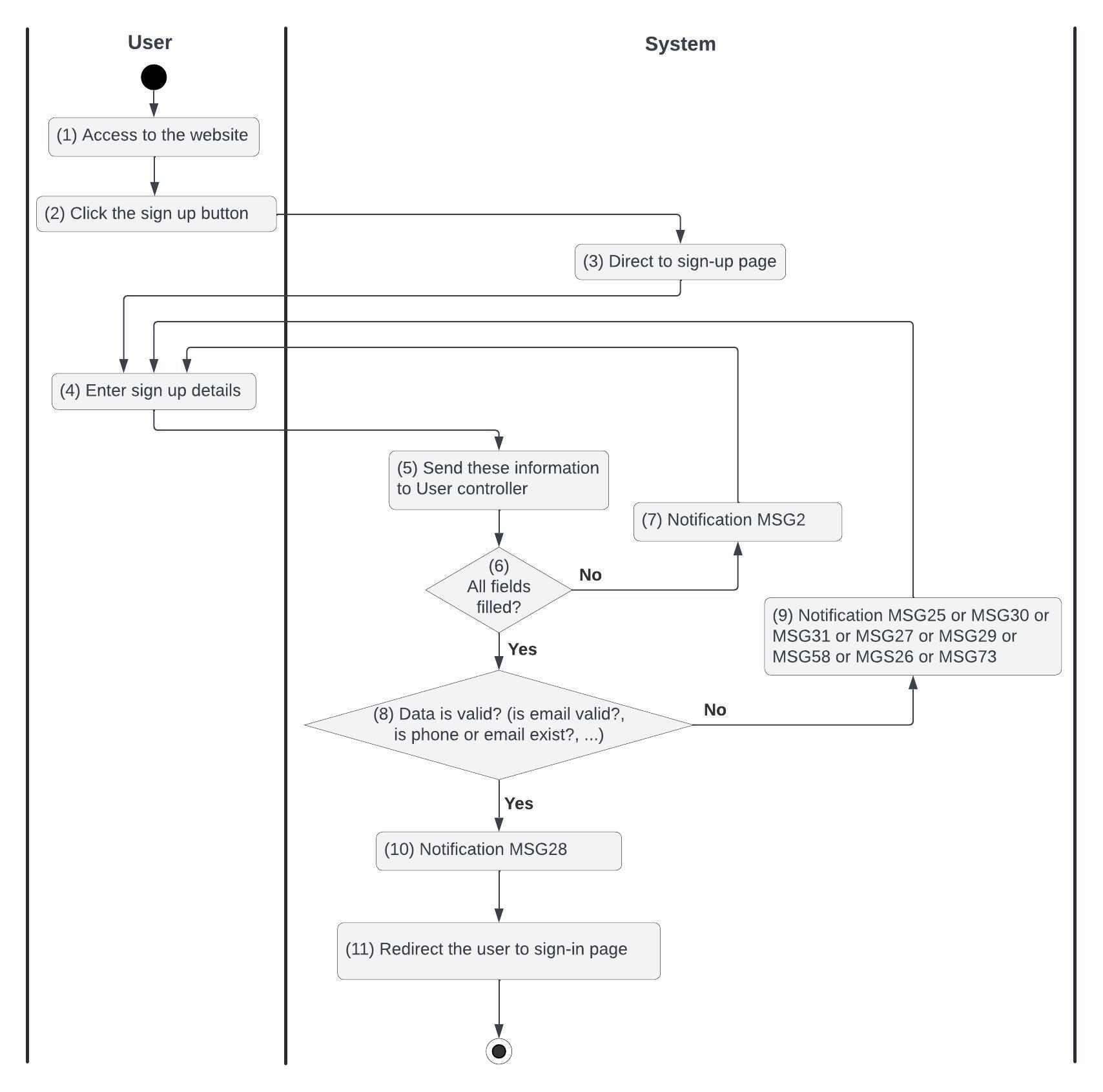
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR1* | **Direct Rules:**  ❖ The system directs the user to the sign-in page (refer to the ‘Sign In Form’ in the ‘View Description’ file) |
| *(4)* | *BR2* | **Entering Rules:**  ❖ The user enters their [username] and [password] on the Sign-In page |
| *(6)* | *BR3* | **Validate Rules:**  **❖** The system checks the items IsFilled([username]), IsFilled([password]):   * If any of them is null or blank the system will show the error message MSG2 |
| *(7)* | *BR4* | **Message Rules:**  ❖ The system shows the error message MSG2 |
| *(8)* | *BR5* | **Validate Rules:**  ❖ The system checks the items IsValidUserName([username]), IsValidPassword([password]):   * If DoesUsernameExist([username]) == False does not exist the system will show the error message MSG22 else [user] = User * Repository find by [username] (call findById() function) * If hash([password]) != user.password then the system will show the error message MSG22 * Else generate JSON Web Token from [user.id], record this login session, and show the success message MSG23 |
| *(9)* | *BR6* | **Message Rules:**  ❖ The system shows the error message MSG22 |
| *(10)* | *BR7* | **Message Rules:**  ❖ The system shows the success message MSG23 |
| *(11)* | *BR8* | **Redirect Rules:**  ❖ The system redirects the user to the home page |

### UC2: Sign Up

| **Name** | **Sign Up** |
| --- | --- |
| **Description** | This use case describes the process by which a user creates a new account in the system |
| **Actor** | User |
| **Trigger** | * When a user clicks the ‘Sign up’ button |
| **Pre-condition** | * The system should be online and accessible * The user is not logged into the system and has internet access * The sign-up button is clickable and actionable and then redirects the user to the sign-up page * The user is on the sign-up page (refer to the ‘Sign Up Form’ in the ‘View Description’ file) |
| **Post-condition** | * A new account will be created in the ‘INACTIVE’ state * The user will be redirected to the sign-in page * The user will be asked to verify through email |

#### Activities Flow

**

*Figure 2: Activities Flow of the sign-up use case*

#### Business Rules

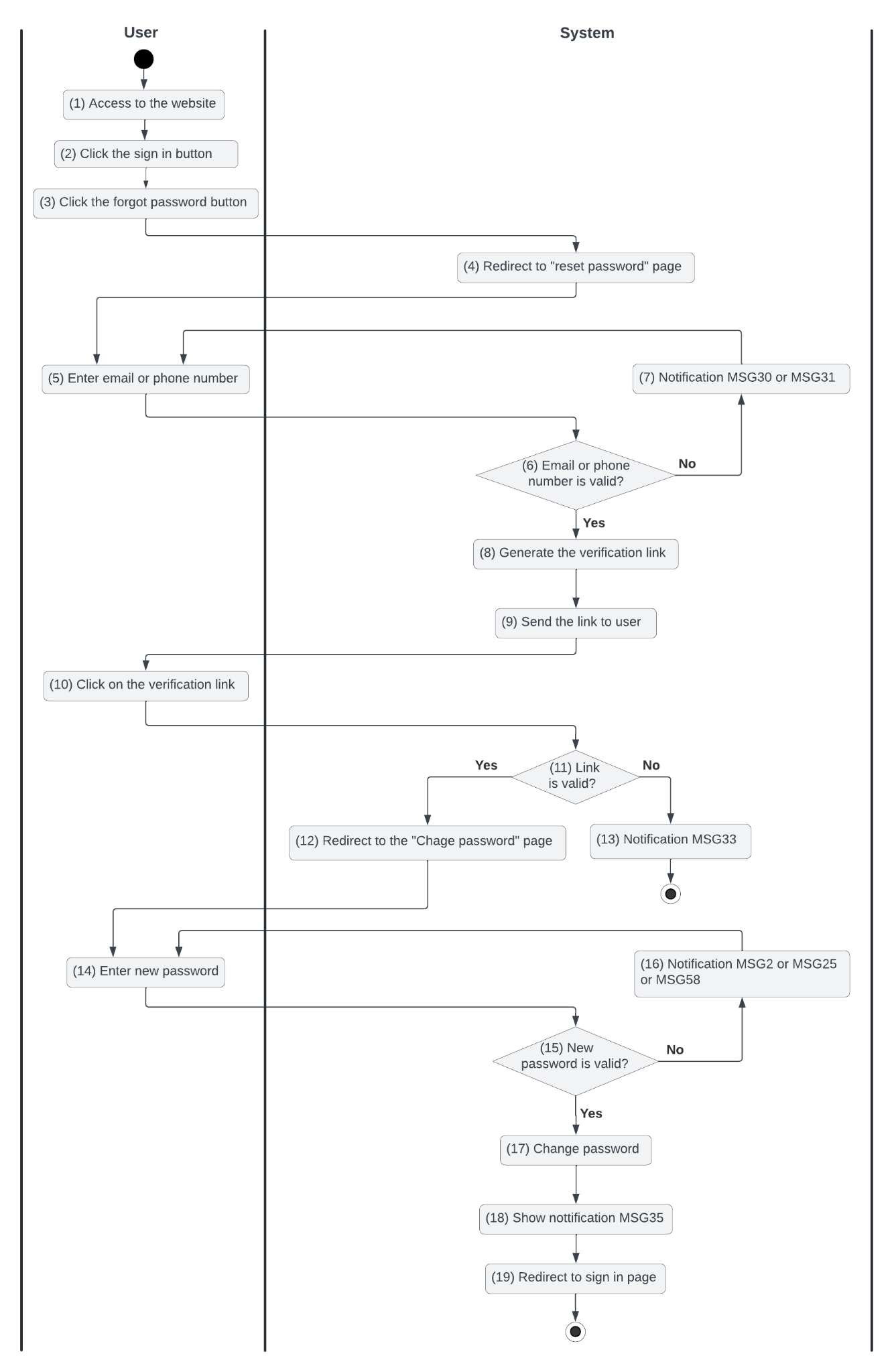
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR9* | **Direct Rules:**  ❖ The system directs the user to the sign-up page (refer to the ‘Sign Up Form’ in the ‘View Description’ file) |
| *(4)* | *BR10* | **Entering Rules:**  ❖ The user enters their [username], [password], [rePassword], [phoneNumber], and [email] on the Sign-up page |
| *(6)* | *BR11* | **Validate Rules:**  ❖ The system checks the items IsFilled([username]), IsFilled([password]), IsFilled([rePassword], IsFilled([phoneNumber]), IsFilled([email]):   * If any entries are empty, the system shows an error message MSG2 |
| *(7)* | *BR12* | **Message Rules:**  ❖ The system shows the error message MSG2 |
| *(8)* | *BR13* | **Validate Rules:**  ❖ The system checks the items IsValidUserName([username]), IsValidPassword([password]), IsValidRePassword([rePassword]), IsValidPhoneNumber([phoneNumber]), IsValidEmail([email]):   * If [username.length] < 8 then the system shows an error message MSG24 * If DoesUsernameExist([username]) == True exists in the system then the system shows an error message MSG73 * If pattern.compile(“" ^ (?=.\*[0-9]) (?=.\*[a-zA-Z]) (?=.\*[@#$%^&+=!]) (?=\\S+$).{8,}$ "”).notMatch([password]) then the system shows an error message MSG25 * If [password] != [rePassword], the system shows an error message MSG58 * If pattern.compile (‘ ^(84|0[3|5|7|8|9]) + ([0-9]{8})$ ’).notMatch([ phoneNumber]) or then returns 400-BAD\_REQUEST error with error message MSG30 * If pattern.compile(‘^ [a-zA-Z0-9.\_%+-] + @[a-zA-Z0-9.-] + \.[a-zA-Z]{2,}$ ’).notMatch([email]) then returns 400-BAD\_REQUEST error with error message MSG31. * If DoesPhoneNumberExist([phoneNumber]) == True exists in the system then the system shows an error message MSG26 * If DoesEmailExist([email]) == True exists in the system then the system shows an error message MSG27 * Else [user] = User Repository save new user with all data (call save() function), [user.status] = ‘INACTIVE’. The system will show a success message MSG28 and Send verify email as **Email Templates** below,   ❖  **Email Templates:**   * Send an email to the user-registered account as the template below:  | From | mbx@gmail.com | | --- | --- | | To | [email] | | Cc | N/A | | Subject | Get [Subject] of “Email Template” item of which [Keyword] = “Sign Up” | | Body | Get [Body] of “Email Template” item of which [Keyword] = “Sign Up” |     ❖ Following is sample email content:   | Subject | "Verify Registration MBX Account" | | --- | --- | | Body | [Body] = “Hello,”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “Follow this link to verify your email address to finish your registration step.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + <<Link to verify email>>  [Body] = [Body] + "If you didn’t ask to verify this address, you can ignore this email."  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Thanks."  [Body] = [Body] + 2 new lines  [Body] = [Body] + "The MBX team" | |
| *(9)* | *BR14* | **Message Rules:**  ❖The system shows the error message MSG25 or MSG30 or MSG31 or MSG27 or MSG29 or MSG58 |
| *(10)* | *BR15* | **Message Rules:**  ❖The system shows the success message MSG28 |
| *(11)* | *BR16* | **Redirect Rules:**  ❖The system redirects the user to the sign-in page |

### 

### UC3: Forgot Password

| **Name** | **Forgot Password** |
| --- | --- |
| **Description** | This use case describes the process by which users reset their password when they forget it |
| **Actor** | User |
| **Trigger** | * When a user clicks the "Forgot Password" button on the sign-in page |
| **Pre-condition** | * The system should be online and accessible * The user is not logged into the system and has internet access * The user is on the sign-in page * The forgot password button is clickable and actionable and then redirects the user to the forgot password page * The user is on the forgot password page (refer to “Forgot Password Form” in the “List Description” file) |
| **Post-condition** | * The user receives a confirmation message for the password reset request * The password has been changed. * The user is redirected to the sign-in page after the password reset |

#### Activities Flow

**

*Figure 3: Activities Flow of the forgot password use case*

#### Business Rules

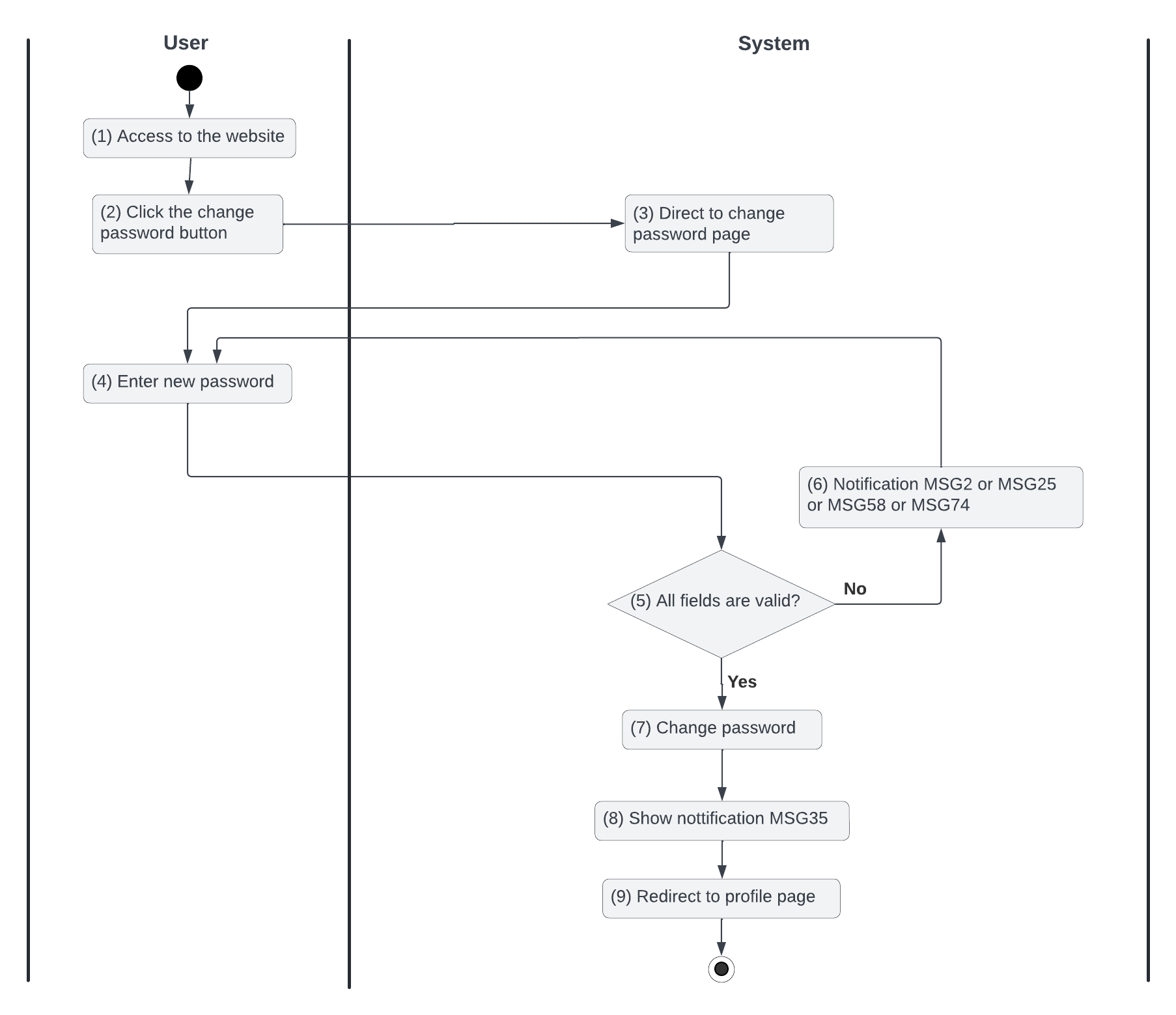
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(4)* | *BR17* | **Direct Rules:**  The system redirects the user to the forgot password page (refer to “Forgot Password Form” in the “List Description” file) |
| *(5)* | *BR18* | **Entering Rules:**  ❖ The user enters their [phoneNumber] or [email] that they have registered by |
| *(6)* | *BR19* | **Validate Rules:**  ❖ The system will checks the items IsValidEmail([email]) or IsValidPhoneNumber([phoneNumber]):   * If *IsFilled([email])* == False or IsFilled([phoneNumber]) == False is null or blank, return the 400-BAD\_REQUEST error with the error message MSG2 * If pattern.compile(‘^(84|0[3|5|7|8|9])+([0-9]{8})$’).notMatch([phoneNumber]) or then returns 400-BAD\_REQUEST error with error message MSG30 * If pattern.compile(‘^ [a-zA-Z0-9.\_%+-] + @[a-zA-Z0-9.-] + \.[a-zA-Z]{2,}$ ’).notMatch([email]) then returns 400-BAD\_REQUEST error with the error message MSG31   ❖ [user] = User Repository findByPhoneOrEmail([phoneNumber], [email]) (call findByPhoneOrEmail() function)   * If [user] == null then returns a 400-BAD\_REQUEST error with the error message MSG32. |
| *(7)* | *BR20* | **Message Rules:**  ❖ The system shows the error message MSG30 or MSG31 |
| *(8)* | *BR21* | **Generate Link Rules:**  ❖ **[**link.expired\_in] = Date.now().plus(10, MINUTES) |
| *(9)* | *BR22* | **Send Link Rules:**  ❖ **Email Templates:**  ❖ Send mail to the user to change the password or message through zalo as the template below:   | From | MBX@gmail.com | | --- | --- | | To | [user.email] | | Cc | N/A | | Subject | Get [Subject] of “Email Template” item of which [Keyword] = “Forgot Password” | | Body | Get [Body] of “Email Template” item of which [Keyword] = “Forgot Password” |   **Following is sample email content:**   | Subject | "Reset Password" | | --- | --- | | Body | [Body] = “Hello,”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “You have requested to reset the password of your MBX account.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Please click the link to change your password: "  [Body] = [Body] + 2 new lines  [Body] = [Body] + [link]  [Body] = [Body] + 2 new lines  [Body] = [Body] + "If you didn’t ask to reset your password, you can ignore this email."  [Body] = [Body] + "Thanks."  [Body] = [Body] + 2 new lines  [Body] = [Body] + "The MBX team" | |
| *(11)* | *BR23* | **Validate Rules:**  ❖ If [link.expired\_in].isAfter(Date,now()) then the system redirects the user to the Reset Password (refer to the “Reset Password” form in the “List Description” file)  else the system will show an error message MSG33 |
| *(12)* | *BR24* | **Redirect Rules:**  ❖ The system redirects the user to the reset password page (refer to the “Reset Password” form in the “List Description” file) |
| *(13)* | *BR25* | **Message Rules:**  ❖ The system shows the error message MSG33 |
| *(14)* | *BR26* | **Entering Rules:**  ❖ The user enters their new [pasword] and [rePassword] |
| *(15)* | *BR27* | **Validate Rules:**  ❖ The system will check the item IsValidPassword([password]) and IsValidRePassword([rePassword]):   * If IsFilled([password]) == False is null or blank, return the 400-BAD\_REQUEST error with the error message MSG2. * If pattern.compile(“"^ (?=.\*[0-9]) (?=.\*[a-zA-Z]) (?=.\*[@#$%^&+=!]) (?=\\S+$).{8,}$"”).notMatch([password]) then the system shows an error message MSG25. * If [password] != [rePassword], the system shows an error message MSG58   Else [user.password] = hash([password]), then the system shows a success message MSG35 |
| *(16)* | *BR28* | **Message Rules:**  ❖ The system shows the error message MSG2 or MSG25 or MSG58 |
| *(18)* | *BR29* | **Message Rules:**  ❖ The system shows the success message MSG35 |
| *(19)* | *BR30* | **Redirect Rules:**  ❖ The system redirects the user to the sign-in page |

### 

### UC4: Change Password

| **Name** | **Change Password** |
| --- | --- |
| **Description** | This use case describes the process by which users can change their password |
| **Actor** | User |
| **Trigger** | * When a user clicks the "Change Password" button on the account dropdown |
| **Pre-condition** | * The system should be online and accessible * The user is logged into the system and has internet access * The change password button is clickable and actionable and then redirects the user to the change password page * The user is on the change password page (refer to “Change Password” in the “List Description” file) |
| **Post-condition** | * The password has been changed. * The user is redirected to the profile page after the password reset |

#### Activities Flow



*Figure 4: Activities Flow of the change password use case*

#### Business Rules

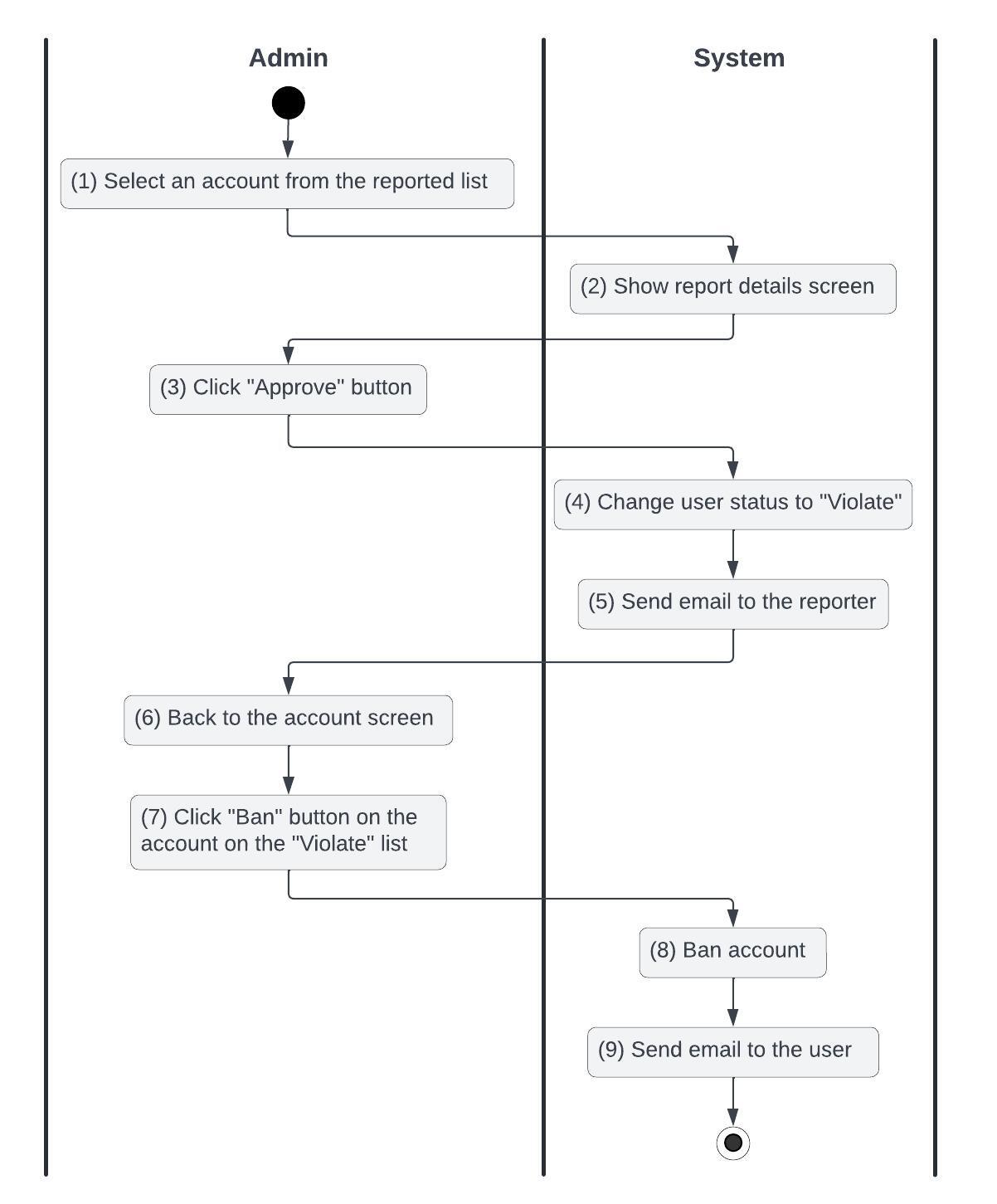
| **Activity** |  |  |
| --- | --- | --- |
| *(3)* | *BR31* | **Redirect Rules:**  ❖ The system redirects the user to the change password page (refer to the “Change Password” form in the “List Description” file) |
| *(4)* | *BR32* | **Entering Rules:**  ❖ The user enters their [old\_password], [new\_password] and [rePassword] |
| *(5)* | *BR33* | **Validate Rules:**  ❖ The system will check the item isValidOldPassword([old\_password], IsValidPassword([new\_password]) and IsValidRePassword([rePassword]):   * If IsFilled([old\_password]) == False, IsFilled([new\_password]) == False or IsFilled([rePassword]) == False is null or blank, return the error message MSG2. * if old\_password != user.password, return the error message MSG74 * If pattern.compile(“"^ (?=.\*[0-9]) (?=.\*[a-zA-Z]) (?=.\*[@#$%^&+=!]) (?=\\S+$).{8,}$"”).notMatch([new\_password]) then the system shows an error message MSG25. * If [password] != [rePassword], the system shows an error message MSG58 * If hash([new\_password]) == [user.password] then the system shows an error message MSG34.   Else [user.password] = hash([new\_password]), then the system shows a success message MSG35 |
| *(6)* | *BR34* | **Message Rules:**  ❖ The system shows the error message MSG2 or MSG25 or MSG34 or MSG58 or MSG74 |
| *(8)* | *BR35* | **Message Rules:**  ❖ The system shows the success message MSG35 |
| *(9)* | *BR36* | **Redirect Rules:**  ❖ The system redirects the user to the profile page |

### 

### UC5: Ban an account

| **Name** | **Ban an account** |
| --- | --- |
| **Description** | This use case describes how a user can ban a user account that violates the website’s regulations |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Ban Account” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The ban account button is clickable and actionable * The Admin accessed the user screen * The user account has been reported in violation |
| **Post-condition** | * The user account is banned |

#### Activities Flow



*Figure 5: Activities Flow of the ban an account use case*

#### Business Rules

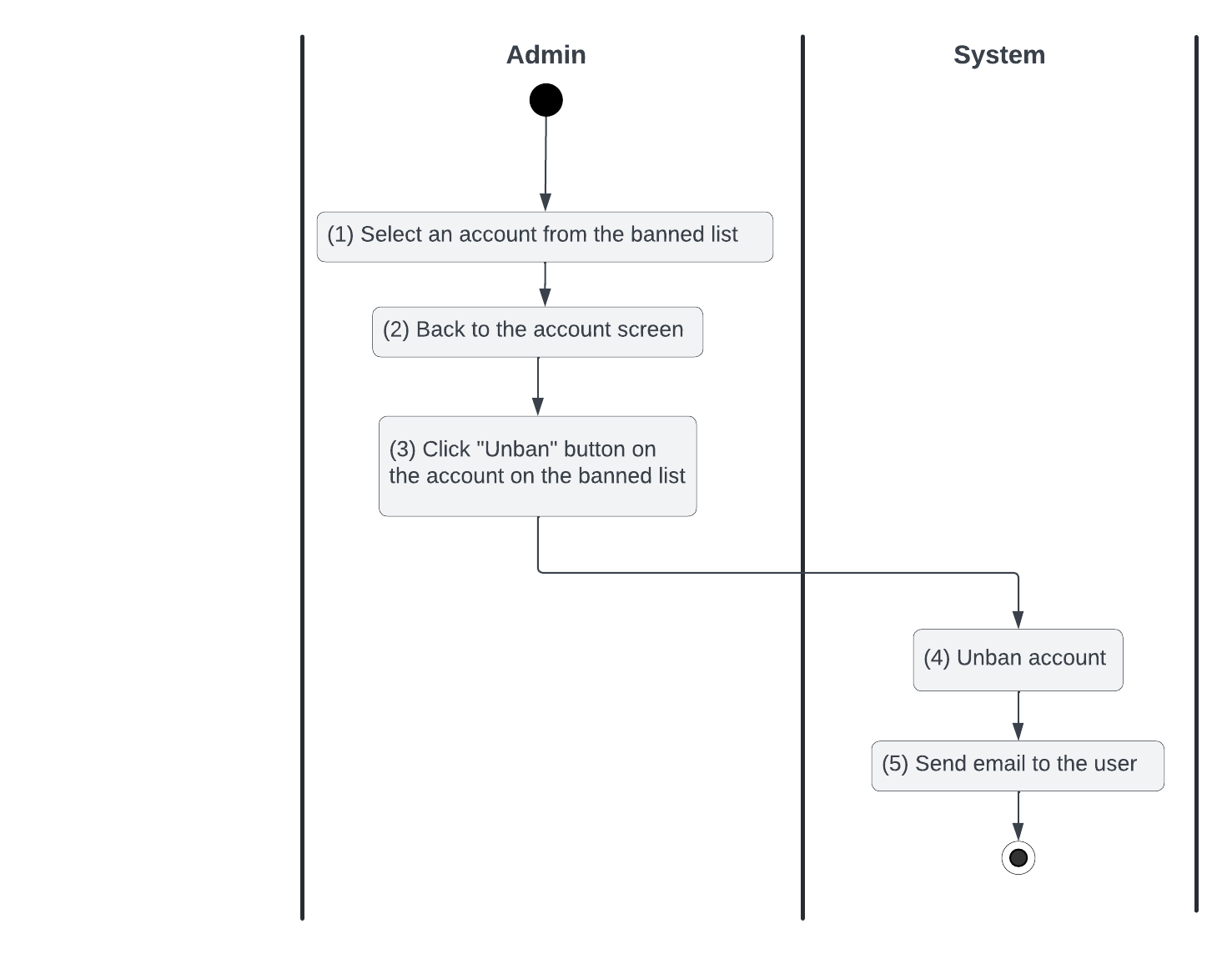
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR37* | **Loading Screen Rules:**  ❖ The system loads the “Report details” screen |
| *(4)* | *BR38* | **Changing Rules:**  ❖ [user.status] = VIOLATE |
| *(5)* | *BR39* | **Sending email rules:**  ❖ Email Templates:  ❖ Send mail to the user after approving their report:   | From | MBX@gmail.com | | --- | --- | | To | [user.email] | | Cc | N/A | | Subject | Get [Subject] of “Email Template” item of which [Keyword] = “Your report has been approved” | | Body | Get [Body] of “Email Template” item of which [Keyword] = “Your report has been approved” |  * Following is a sample email content:  | Subject | "Your report has been approved - MBX" | | --- | --- | | Body | [Body] = “Hello,”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “Thanks for your report on the post [post.title]!”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “We have reviewed and confirmed the user [user.username] violated our rules. Posts by this person will be taken down.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Thank you for your cooperation. Hope you continue to accompany us.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Best regards,"  [Body] = [Body] + 2 new lines  [Body] = [Body] + "The MBX team" | |
| *(8)* | *BR40* | **Ban Rules:**  ❖ [user.status] = BANNED |
| *(9)* | *BR41* | **Sending email rules:**  ❖ Email Templates:  ❖ Send mail to the user after changing their account status:   | From | MBX@gmail.com | | --- | --- | | To | [user.email] | | Cc | N/A | | Subject | Get [Subject] of “Email Template” item of which [Keyword] = “Account violates regulations” | | Body | Get [Body] of “Email Template” item of which [Keyword] = “Account violates regulations” |  * Following is a sample email content:  | Subject | "Account violates regulations - MBX" | | --- | --- | | Body | [Body] = “Hello,”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “We found that your account violates our posting rules.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “Report Details:”  [Body] = [Body] + 1 new lines  [Body] = [Body] + “- Reason: [report.reason]”  [Body] = [Body] + “- Details: [report.description]”  [Body] = [Body] + "If you have any questions or concerns, please feel free to contact our customer support."  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Thank you for using our service!"  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Best regards,"  [Body] = [Body] + 2 new lines  [Body] = [Body] + "The MBX team" | |

### 

### UC6:Unban an account

| **Name** | **Unban an account** |
| --- | --- |
| **Description** | This use case describes how a user can unban a user account |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Unban Account” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The unban account button is clickable and actionable * The Admin accessed the user screen |
| **Post-condition** | * The user account is unbanned |

#### Activities Flow



*Figure 6: Activities Flow of the unban an account use case*

#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(4)* | *BR42* | **Ban Rules:**  ❖ [user.status] = Active |
| *(5)* | *BR43* | **Sending email rules:**  ❖ Email Templates:  ❖ Send mail to the user after changing their account status:   | From | MBX@gmail.com | | --- | --- | | To | [user.email] | | Cc | N/A | | Subject | Get [Subject] of “Email Template” item of which [Keyword] = “Unban Account” | | Body | Get [Body] of “Email Template” item of which [Keyword] = “Unban Accounts” |  * Following is a sample email content:  | Subject | "Unban Account- MBX" | | --- | --- | | Body | [Body] = “Hello,”  [Body] = [Body] + 2 new lines  [Body] = [Body] + “We have unbanned your account.”  [Body] = [Body] + 2 new lines  [Body] = [Body] + "If you have any questions or concerns, please feel free to contact our customer support."  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Thank you for using our service!"  [Body] = [Body] + 2 new lines  [Body] = [Body] + "Best regards,"  [Body] = [Body] + 2 new lines  [Body] = [Body] + "The MBX team" | |

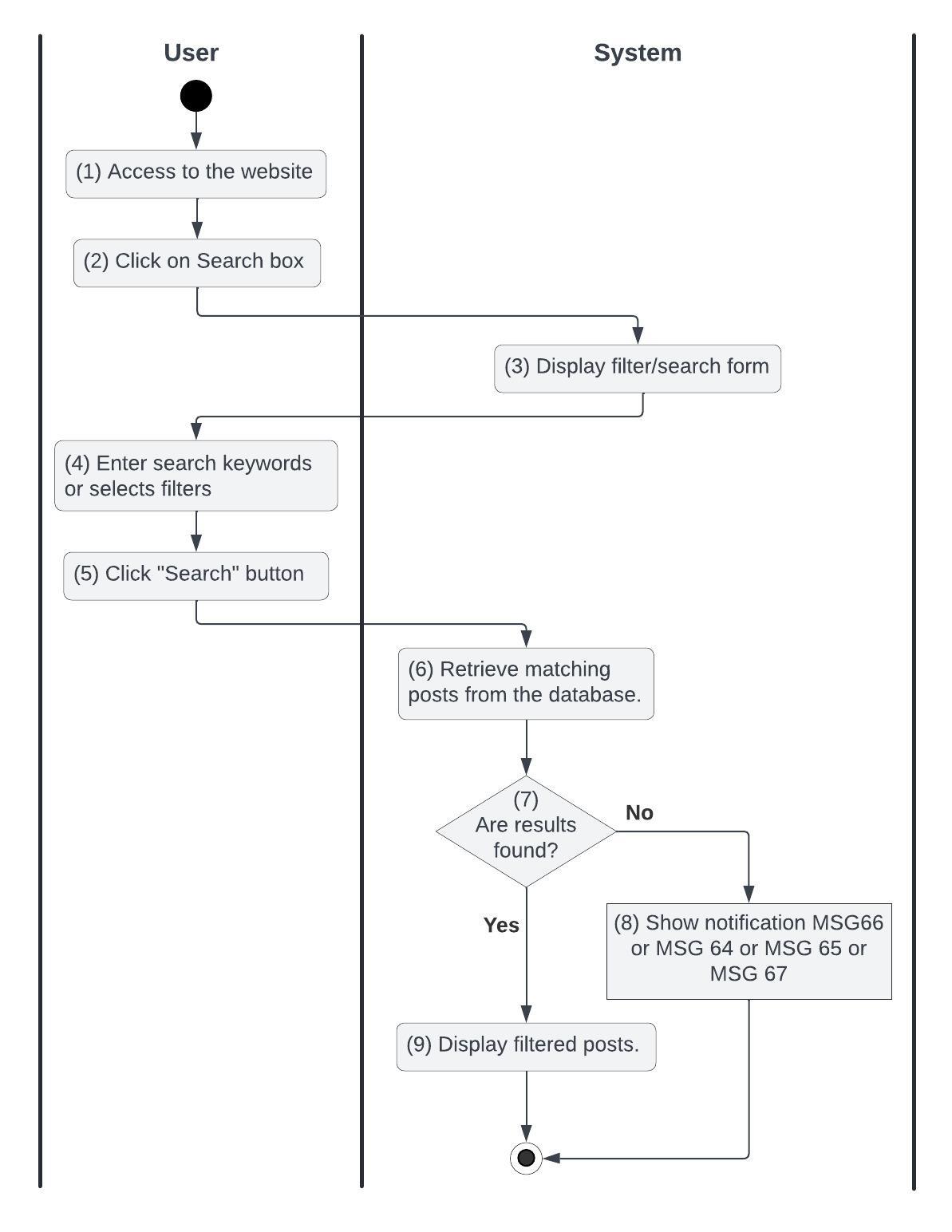
### 

### 

### UC7: Filter/Search post

| **Name** | **Filter/Seach Post** |
| --- | --- |
| **Description** | This use case allows users to filter and search for posts based on specific criteria such as keywords, categories, location, and price range |
| **Actor** | User |
| **Trigger** | * When a user enters search keywords or selects filtering options and clicks the "Search" button |
| **Pre-condition** | * The system should be online and accessible * The user has internet access * Posts must be available in the system for filtering and searching * The search button is clickable and actionable * The search bar and filter options are displayed and functional * The user is on the filter/search form (refer to the “Filter/Search” form in the “List Description” file) |
| **Post-condition** | * The system displays a list of posts that match the specified search or filter criteria. * If no results match, the system displays an error message |

#### Activities Flow



*Figure 7: Activities Flow of the filter/search use case*

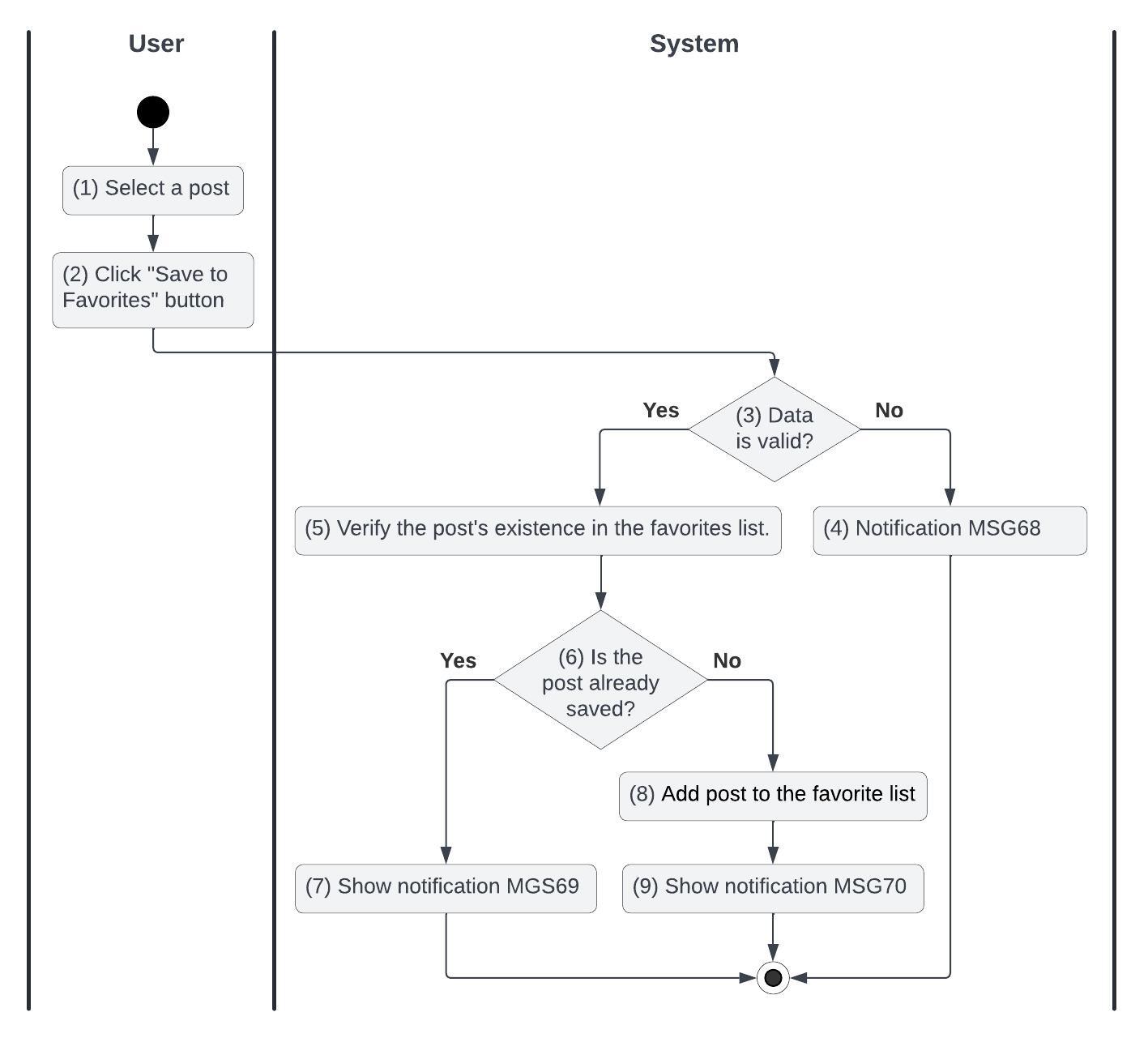
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR44* | **Direct Rules:**  ❖ The system redirects the user to the filter/search form (refer to “Filter/Search” form in the “List Description” file) |
| *(4)* | *BR45* | **Entering Rules:**  ❖ The user enters keywords or selects filtering options, including: [keywords], [categories], [priceRange], [province], [district], [ward] |
| *(7)* | *BR46* | **Retrieving Rules:**  ❖ The system concatenates values from [province], [district], and [ward] to form [area]  ❖ The system validates the entered criteria:   * If IsFilled([keywords]) == False and no filters are selected, the system shows an error message MSG64 * If [price range.min] < 0 or [price range.max] < 0, the system shows an error message MSG65 * If [price range.min] > [price range.max], the system shows an error message MSG66   + The system retrieves posts that match the criteria (by Filter\_Search() function)  ❖ If the Filter\_Search function returns False, the systems show an error message MSG67   * The system displays the filtered/search results in a paginated view, ordered by relevance or date |
| *(8)* | *BR47* | **Message Rules:**  ❖ The system shows the error message MSG66 or MSG 64 or MSG 65 or MSG 67 |
| *(9)* | *BR48* | **Display Rules:**  ❖ The system displays the filtered/search results in a paginated view, ordered by relevance or date |

### UC8: Save Post to Favorites

| **Name** | **Save Post to Favorites** |
| --- | --- |
| **Description** | This use case allows users to save posts to their personal favorites list for easier access later |
| **Actor** | User |
| **Trigger** | * When a user clicks the "Save to Favorites" button on a post |
| **Pre-condition** | * The system should be online and accessible * The user is logged into the website and has internet access * Posts must be available in the system for filtering and searching * The save to favorites button is clickable and actionable * The post exists in the system * The user is on the post form (refer to the “User/Post Details” form in the “List Description” file) |
| **Post-condition** | * The post is added to the user's favorites list |

#### Activities Flow



*Figure 8: Activities Flow of the save post to favorites use case*

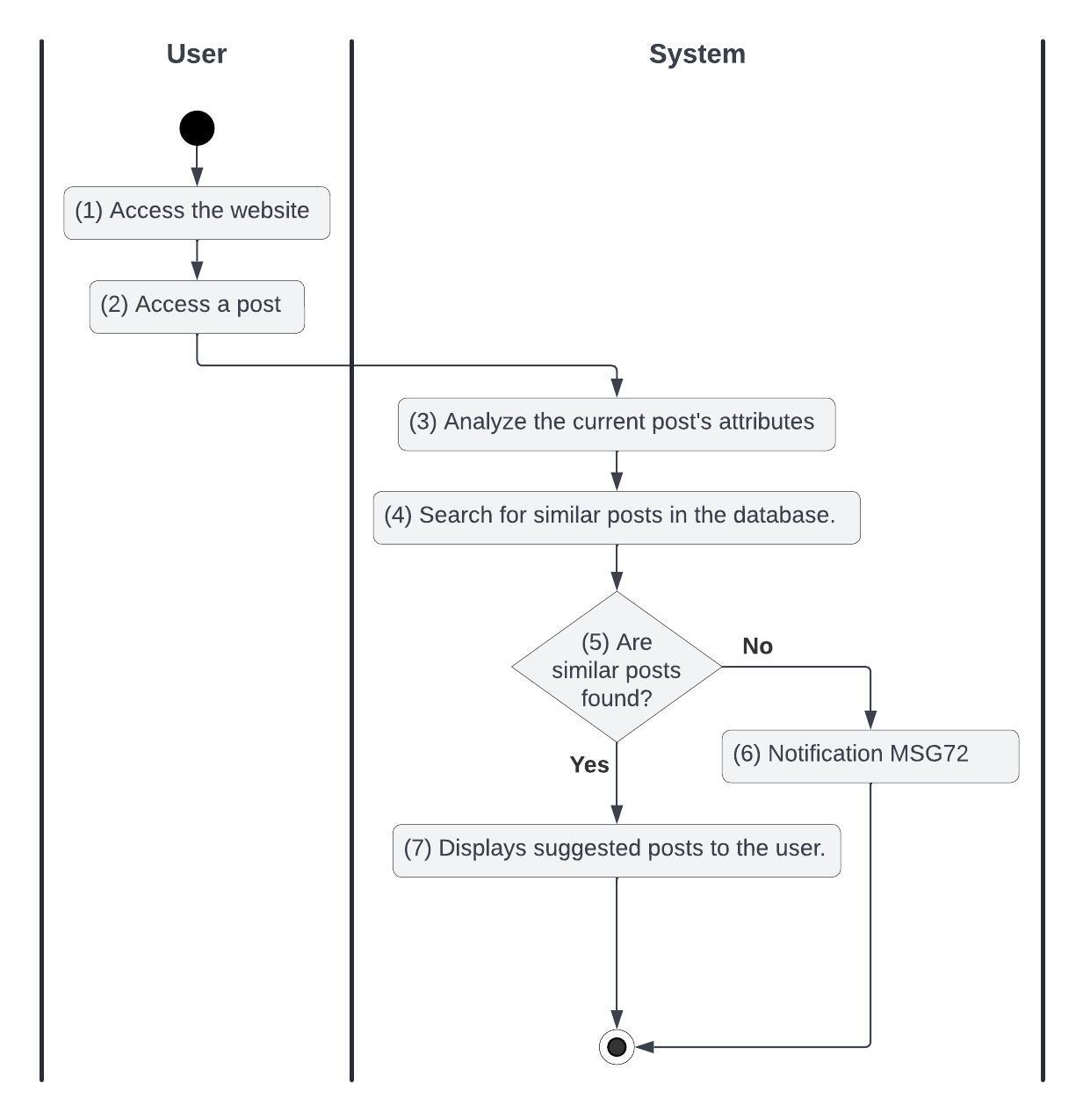
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR49* | **Validate Rules:**  ❖ The system validates the following:   * Check if the post exists in the system using DoesPostExist([postId]). If not, display MSG11 |
| *(4)* | *BR50* | **Message Rules:**  ❖ The system shows an error message MSG68 |
| *(6)* | *BR51* | **Checking Rules:**  ❖ If the post is already in favorites, display MSG69   * Else save the post to favorites, and display MSG70 |
| *(7)* | *BR52* | **Message Rules:**  ❖ The system shows an error message MSG69 |
| *(8)* | *BR53* | **Saving Rules:**  ❖ The system saves the post to the user's favorites list and updates the database with the following:   * [userId] = <<current user id from JWT>> * [postId] = <<current post id>> * [savedDate] = <<current date time>> |
| *(9)* | *BR54* | **Message Rules:**  ❖ The system shows a message MSG70 |

### UC9: Suggest Similar Posts

| **Name** | **Suggest Similar Posts** |
| --- | --- |
| **Description** | This use case describes how the system suggests similar posts to the user based on the current post they are viewing |
| **Actor** | User |
| **Trigger** | * When the user views a post's details, the system automatically suggests related posts |
| **Pre-condition** | * The system should be online and accessible * The user has internet access * The user is viewing a specific post * There are other posts in the system with similar characteristics * The posts exist in the system * The user is on the post form (refer to the “User/Post Details” form in the “List Description” file) |
| **Post-condition** | * The system displays a list of suggested posts similar to the currently viewed post. * The user can click on any suggested post to view its details. |

#### Activities Flow



*Figure 9: Activities Flow of the suggest post use case*

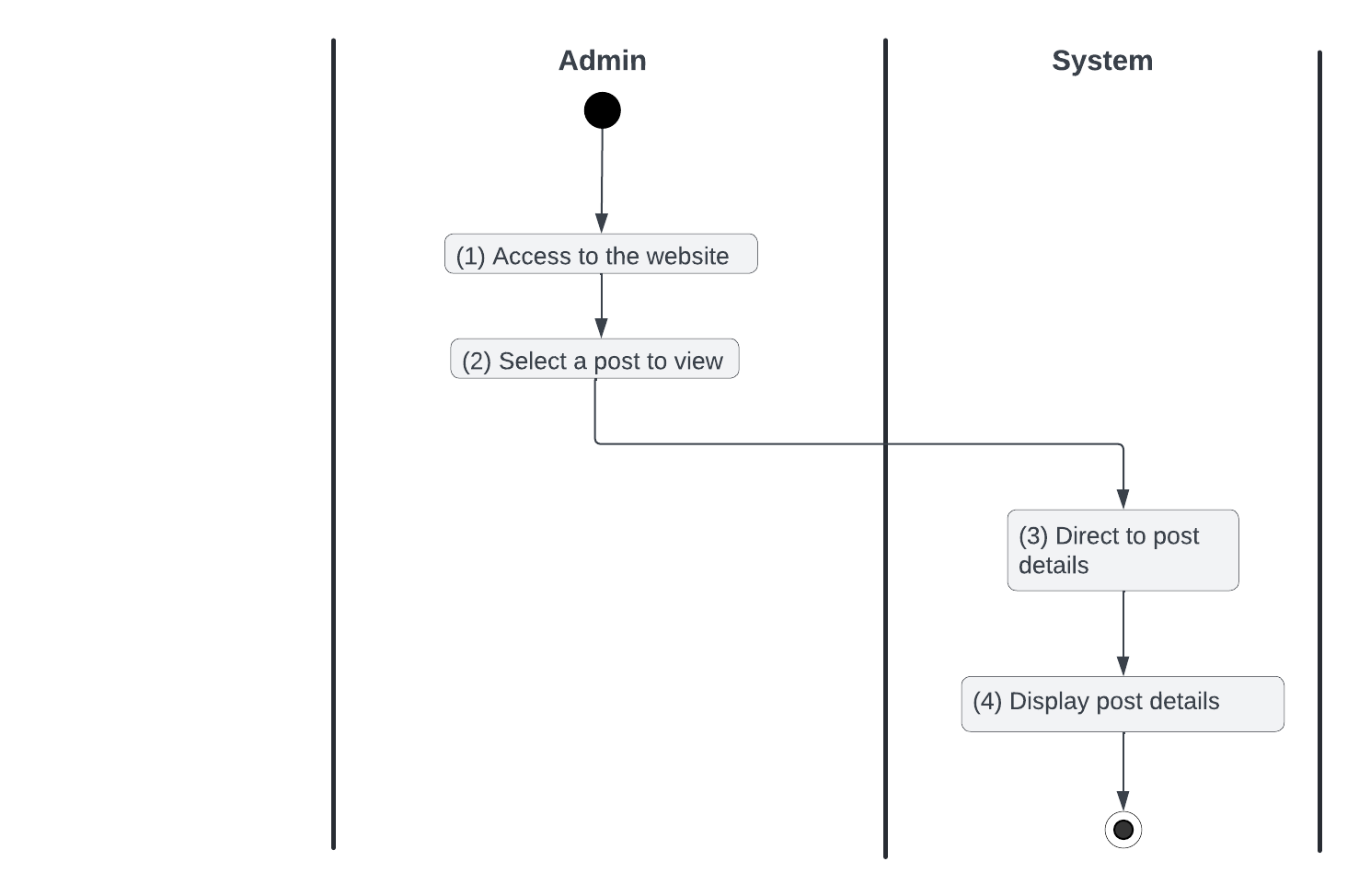
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(4)* | *BR55* | **Retrieving Rules:**  ❖ When a user views a post, the system retrieves the post details, including [product], [category], [price], [province], [district],[ward], [title] and [description].  ❖ The systems search for similar posts by GetSimilarPosts([postId]) function  ❖ The system uses the following criteria to find similar posts:   * Posts with the same [category]. * Posts with [price] within ±20% of the current post's price. * Posts from the same or nearby [province], [district], and [ward] * Posts with similar keywords in [title] or [description].   ❖ The system retrieves up to 10 matching posts using a relevance ranking algorithm. |
| *(5)* | *BR56* | **Checking Rules:**  ❖ If the GeSimilarPosts Function returns None, the system shows the message MSG72   * Else the system shows up to 10 matching posts |
| *(6)* | *BR57* | **Message Rules:**  ❖ The system shows an error message MSG72 |
| *(7)* | *BR58* | **Display Rules:**  ❖ The system displays the retrieved posts in a "Similar Posts" section below the current post details. ❖ Each suggested post shows:   * [image] * [title] * [price] * [province] * [district] * [ward] |

### UC10: View Post

| **Name** | **View Post** |
| --- | --- |
| **Description** | This use case the process by which a user views the details of a post |
| **Actor** | User |
| **Trigger** | * The user selects a post to view from the list of posts |
| **Pre-condition** | * The system should be online and accessible * The user has internet access * The view button is clickable and actionable and then redirects the user to the post details form * At least one post exists in the system * The user is on the post details form (refer to “User/Post Details” in the “List Description” file) |
| **Post-condition** | * The system displays the post details |

#### Activities Flow



*Figure 10: Activities Flow of the view post use case*

#### Business Rules

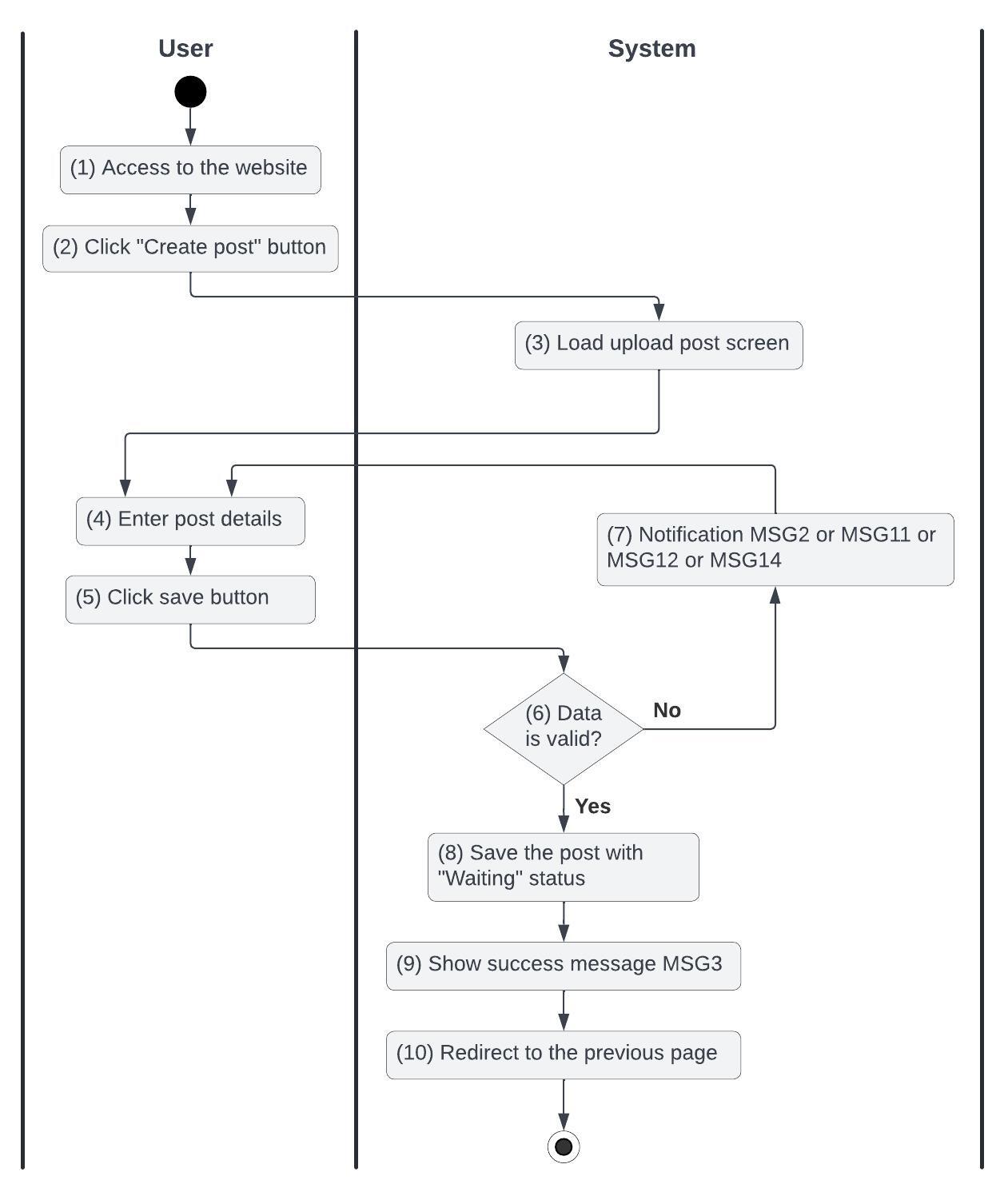
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR59* | **Direct Rules:**  ❖ The system redirects the user to the post details form (refer to “User/Post Details” in the “List Description” file) |
| *(4)* | *BR60* | **Display Rules:**  ❖ the system displays the post details, including:   * [image] * [product] * [price] * [Title] * [description] * Area (concatenation of [province], [distric]t, and [ward]) * Created Date * Post Owner |

### 

### UC11: Create Post

| **Name** | **Create Post** |
| --- | --- |
| **Description** | This use case allows Sellers to create a post that contains their product information |
| **Actor** | Seller |
| **Trigger** | * When a user clicks the "Create Post" button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The create-post button is clickable and actionable and then redirects the user to the create-post form * The user is on the create post form (refer to “Create Post Form” in the “List Description” file) |
| **Post-condition** | * The post has been created * The user is redirected to the previous page that they are before clicking on the create-post button |

#### Activities Flow



*Figure 11: Activities Flow of the create-post use case*

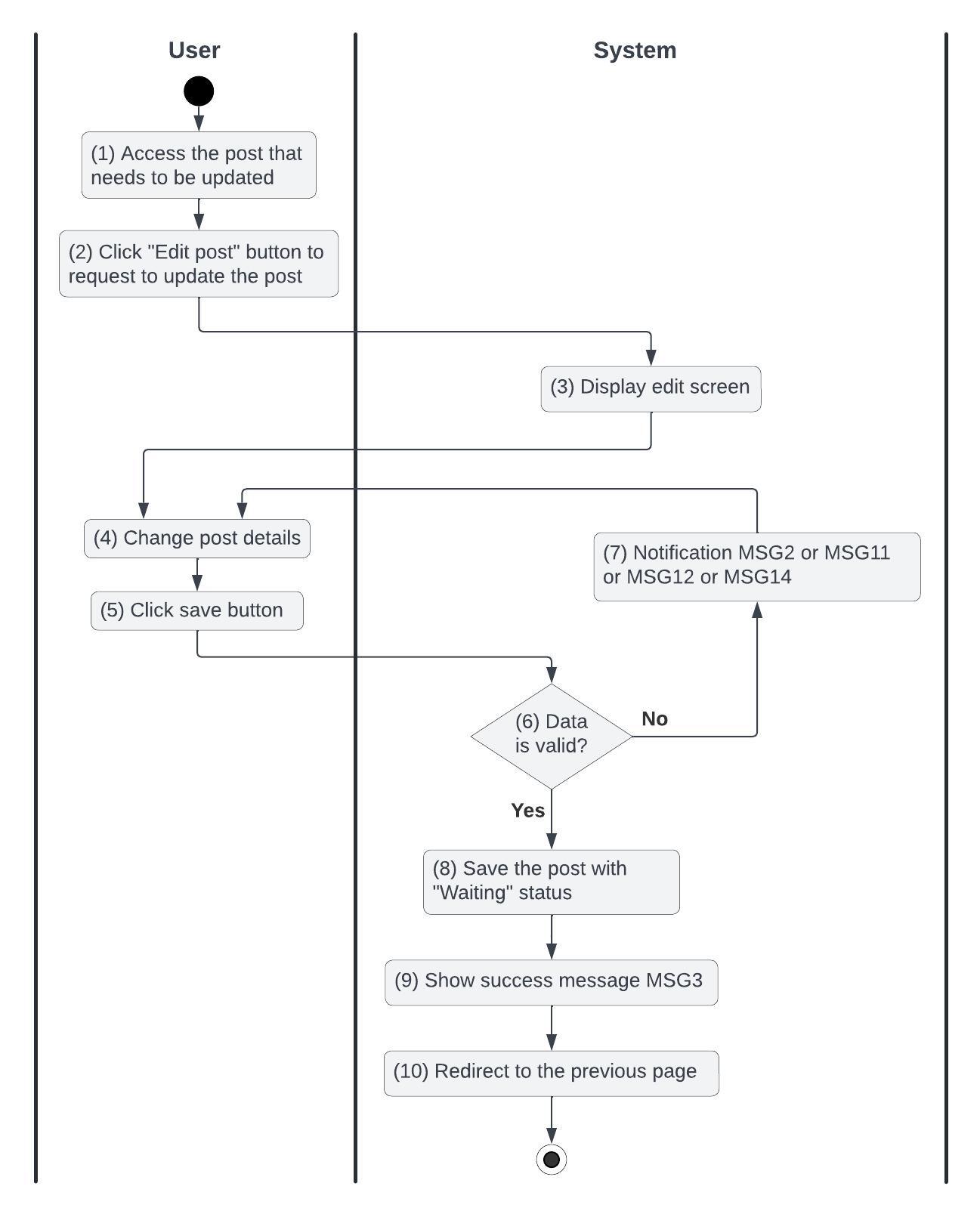
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR61* | **Direct Rules:**  ❖ The system redirects the user to the create post page (refer to “Create Post Form” in the “List Description” file) |
| *(4)* | *BR62* | **Entering Rules:**  ❖ The user enters post details that include: [image], [product], [price], [title], [description], [province], [district], [ward] |
| *(6)* | *BR63* | **Creating Rules:**  ❖ When the user clicks the “Save” button, the system will prompt a confirmation message (Refer to MSG1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:   * The client concat values from [province], [district] and [ward] to [area] * The system checks the items IsValidImage([image]), IsValidProduct([product]), IsValidPrice([price]), IsValidTile([title]), IsValidDescription([description]), IsValidArea([area]):   + If IsFilled([image]) == False or IsFilled([product]) == False or IsFilled([price]) == False or IsFilled([title]) == False or IsFilled([description]) == False or IsFilled([area]) == False is empty, the system shows an error message MSG2   + If the size of any [image] > 8.MB then system shows an error message MSG11   + If DoesProductExist([product]) == False does not exist, the system shows an error message MSG12.   + If [price] < 0 or isDigit([price]) != True, the system shows the error message MSG13.   + If [description.length] < 50, the system shows the error message MSG14. * [user] = <<current user id retrieved from jwt>> * [createdDate] = <<current date time>> * [status] = ‘CREATED’. When a post is in this state, no one except the owner and Administrator can see and interact with it, then the system shows the success message MSG3 |
| *(7)* | *BR64* | **Message Rules:**  ❖ The system shows the error message MSG2 or MSG11 or MSG12 or MSG14 |
| *(9)* | *BR65* | **Message Rules:**  ❖ The system shows the success message MSG3 |
| *(10)* | *BR66* | **Redirect Rules:**  ❖ The system redirects the user to the previous page |

### UC12: Update Post

| **Name** | **Update Post** |
| --- | --- |
| **Description** | This use case describes how a seller can update a post |
| **Actor** | Seller |
| **Trigger** | * When a user clicks the "Update Post" button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The user has created this post * The update-post button is clickable and actionable and then redirects the user to the update-post form * The user is on the update-post form (refer to “Update Post Form” in the “List Description” file) |
| **Post-condition** | * The post has been updated * The user is redirected to the previous page that they are before clicking on the update-post button |

#### Activities Flow



*Figure 12: Activities Flow of the update-post use case*

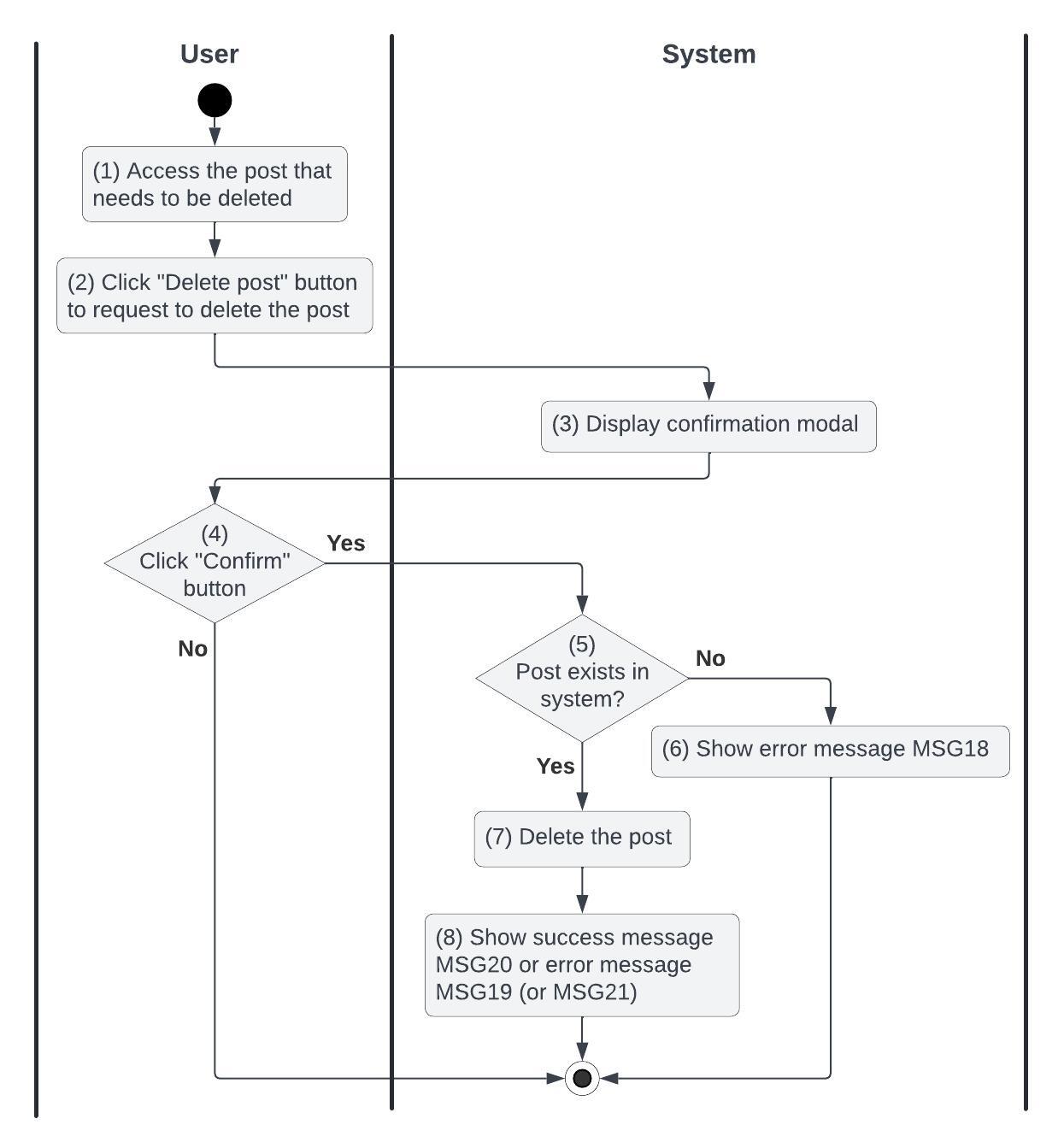
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR67* | **Direct Rules:**  ❖ The system redirects the user to the update-post page (refer to “Update Post Form” in the “List Description” file) |
| *(4)* | *BR68* | **Editing Rules:**  ❖ The user edits post details that include: [image], [product], [price], [title], [description], [area] |
| *(6)* | *BR69* | **Creating Rules:**  ❖ When the user clicks the “Save” button, the system will prompt a confirmation message (Refer to MSG1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:   * The system checks the items IsValidImage([image]), IsValidProduct([product]), IsValidPrice([price]), IsValidTile([title]), IsValidDescription([description]), IsValidArea([area]):   + If IsFilled([image]) == False or IsFilled([product]) == False or IsFilled([price]) == False or IsFilled([title]) == False or IsFilled([description]) == False or IsFilled([area]) == False is empty, the system shows an error message MSG2   + If the size of any [image] > 8.MB then system shows an error message MSG11   + If DoesProductExist[product] == False does not exist, the system shows an error message MSG12.   + If [price] < 0 or isDigit([price]), the system shows the error message MSG13.   + If [description.length] < 50, the system shows the error message MSG14. * [user] = <<current user id retrieved from jwt>> * [createdDate] = <<current date time>> * [status] = ‘CREATED’. When a post is in this state, no one except the owner and Administrator can see and interact with it, then the system shows the success message MSG3 |
| *(7)* | *BR70* | **Message Rules:**  ❖ The system shows the error message MSG2 or MSG11 or MSG12 or MSG14 |
| *(9)* | *BR71* | **Message Rules:**  ❖ The system shows the success message MSG3 |
| *(10)* | *BR72* | **Redirect Rules:**  ❖ The system redirects the user to the previous page |

### UC13: Delete Post

| **Name** | **Delete Post** |
| --- | --- |
| **Description** | This use case describes how a seller can delete a post |
| **Actor** | Seller |
| **Trigger** | * When a user clicks the "Delete Post" button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The user has created this post * The delete-post button is clickable and actionable * The user is on the update-post form (refer to “Update Post Form” in the “List Description” file) |
| **Post-condition** | * The post has been deleted or shows an error message why the post can’t be deleted * The user is redirected to the previous page that they are before clicking on the update-post button |

#### Activities Flow



*Figure 13: Activities Flow of the delete-post use case*

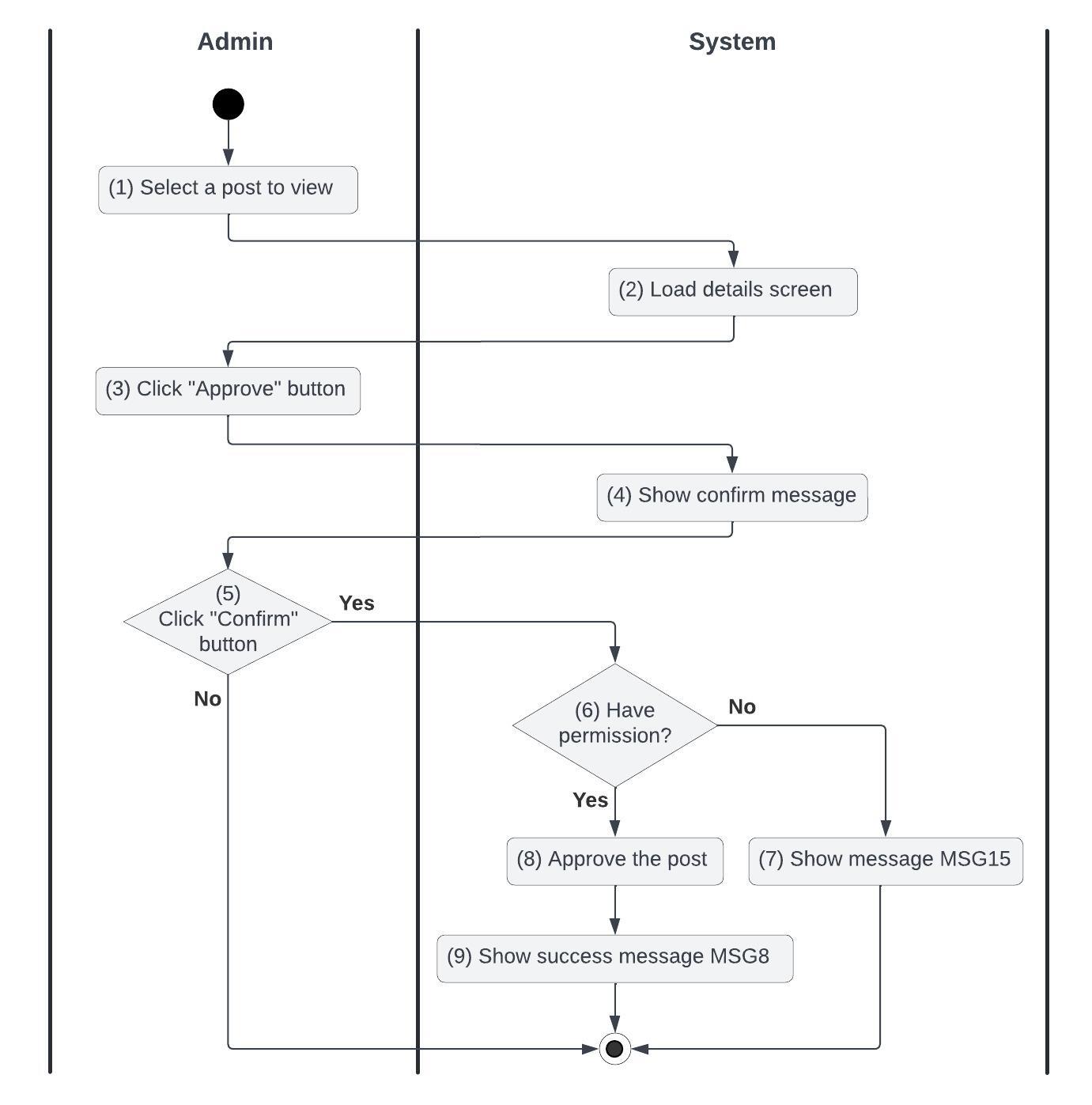
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR73* | **Loading Modal Rules:**  ❖ The system loads the “Delete confirmation” modal |
| *(5)* | *BR74* | **Checking Rules:**  ❖ If DoesPostExist([postId]) does not exist, the system shows an error message MSG18 else [post] = Post Repository find by [postId] (call findById() function) |
| *(6)* | *BR75* | **Message Rules:**  ❖ The system shows the error message MSG18 |
| *(7)* | *BR76* | **Delete Rules:**  ❖ If [post.order] == null then Post Repository delete by [post.id] (call deleteById() function) and shows an success message MSG20  else the system shows an error message MSG19 |
| *(8)* | *BR77* | **Message Rules:**  ❖ The system shows the success message MSG20 or the error message MSG19 (or MSG21) |

### UC14: Approve a Post

| **Name** | **Approve a Post** |
| --- | --- |
| **Description** | This use case allows the Administrator to approve the user's post |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Approve Post” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The approve post button is clickable and actionable * The user is on the “Post details” screen (refer to “Admin/Post Details” in the “List description” file) |
| **Post-condition** | * The post is approved and visible to all users |

#### Activities Flow



*Figure 14: Activities Flow of the approve a post use case*

#### Business Rules

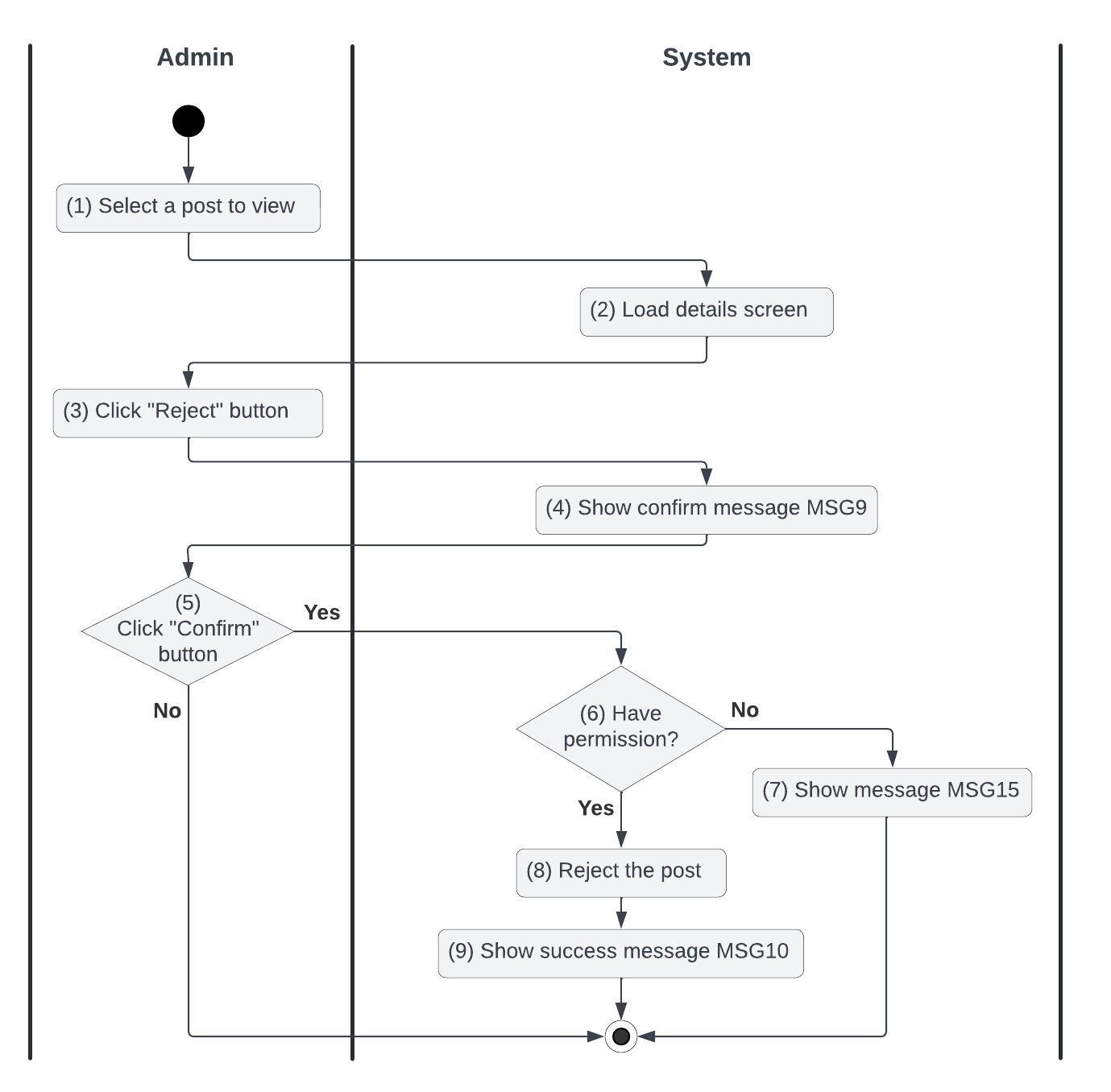
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR78* | **Loading Screen Rules:**  ❖ The system loads the “Post details” screen (refer to “Admin/Post Details” in the “List description” file) |
| *(4)* | *BR79* | **Message Rules:**  ❖ The system shows message MSG7 |
| *(6)* | *BR80* | **Approving rules:**  ❖ If the current user is not ADMIN then the system shows the error message MSG15.  ❖ The system will update the item as the following::   * [postId.status] = APPROVED |
| *(7)* | *BR81* | **Message Rules:**  ❖ The system shows an error message MSG15 |
| *(9)* | *BR82* | **Message Rules:**  ❖ The system shows a success message MSG8 |

### 

### UC15: Reject a Post

| **Name** | **Reject a Post** |
| --- | --- |
| **Description** | This use case allows the Administrator to reject the user's post |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Reject Post” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The reject post button is clickable and actionable * The user is on the “Post details” screen (refer to “Admin/Post Details” in the “List description” file) |
| **Post-condition** | * The post is rejected |

#### Activities Flow



*Figure 15: Activities Flow of the reject a post use case*

#### Business Rules

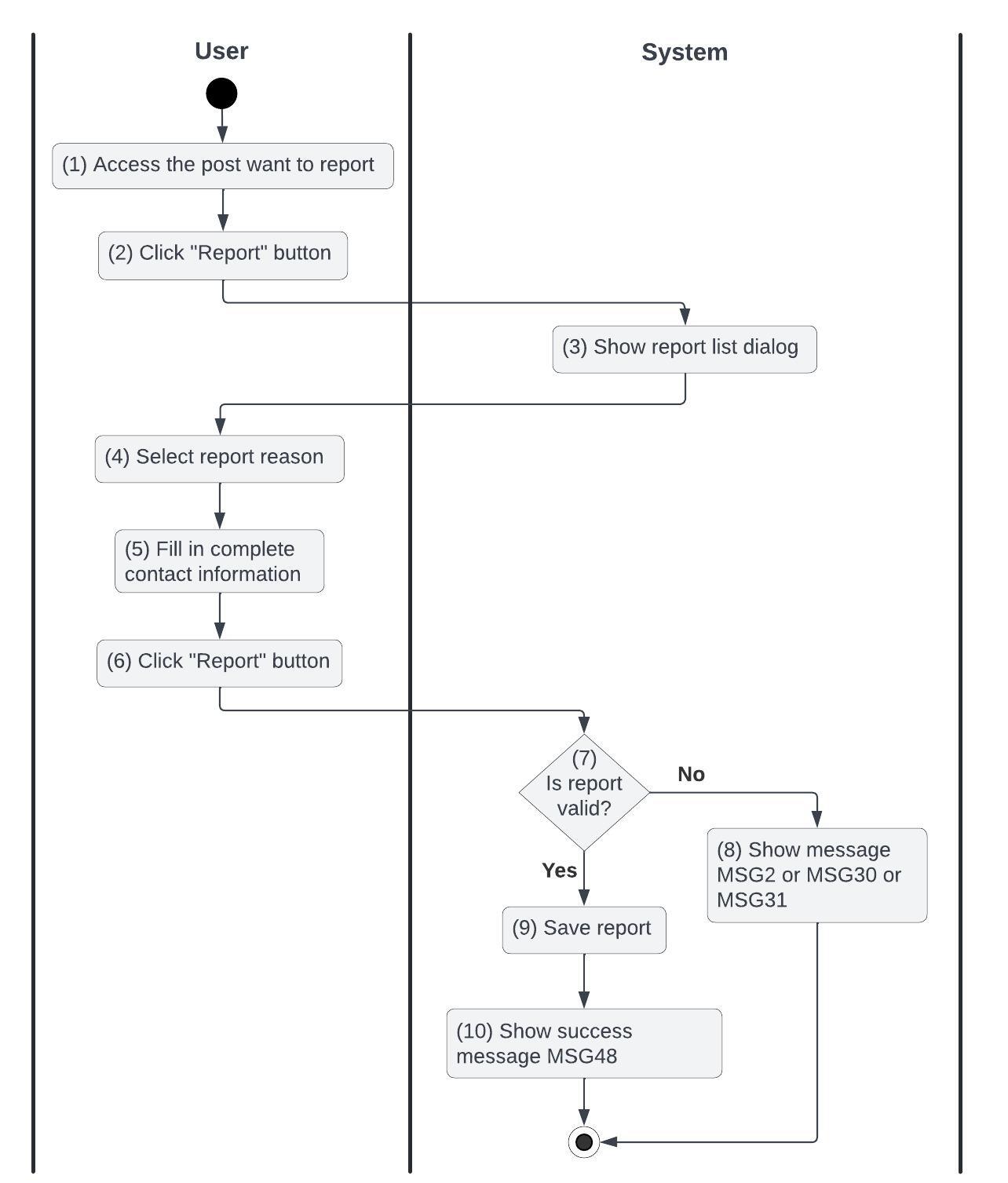
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR83* | **Loading Screen Rules:**  ❖ The system loads the “Post details” screen (refer to “Admin/Post Details” in the “List description” file) |
| *(4)* | *BR84* | **Message Rules:**  ❖ The system shows message MSG9 |
| *(6)* | *BR85* | **Reject rules:**  ❖ If the current user is not ADMIN then the system shows the error message MSG15.  ❖ The system will update the item as the following::   * [postId.status] = DELETED |
| *(7)* | *BR86* | **Message Rules:**  ❖ The system shows an error message MSG15 |
| *(9)* | *BR87* | **Message Rules:**  ❖ The system shows a success message MSG10 |

### 

### UC16: Report a Post

| **Name** | **Report a Post** |
| --- | --- |
| **Description** | This use case describes how users can report a violative post |
| **Actor** | User |
| **Trigger** | * When the user clicks on the ‘Report’ button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The report button is clickable and actionable * The user is on the post details page (refer to “User/Post Details” in the “List description” file) |
| **Post-condition** | * The report is created and sent to Admin for reviewing |

#### Activities Flow



*Figure 16: Activities Flow of the report a post use case*

#### Business Rules

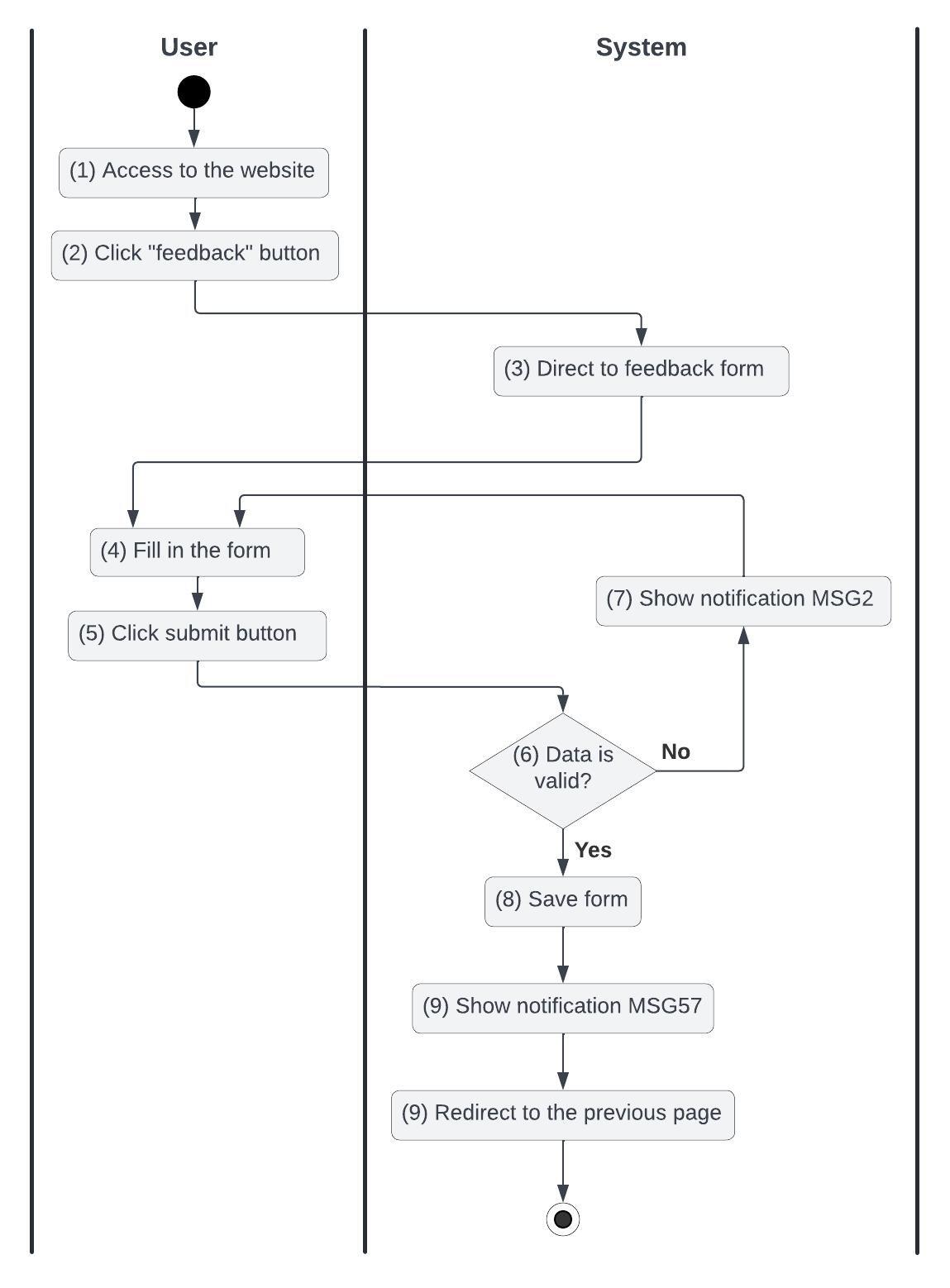
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR88* | **Loading Dialog Rules:**  ❖ The system loads the “Report list” dialog  **Report Reason List Rules:**  ❖ The system shows the following options:   * Cheat * Duplicate * Goods sold * Unable to contact * Information is not factually correct * Counterfeit goods, fake goods * Goods damaged after purchase |
| *(5)* | *BR89* | **Fill in the contact information rules:**  ❖ The system requests the user to provide the following information:   * [report.phoneNumber] * [report.email] * [report.description] |
| *(7)* | *BR90* | **Creating Rules:**  ❖ When the user clicks the “Report” button, the system will prompt a confirmation message (Refer to MSG1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:  ❖ The system checks the following items : [report.phoneNumber], [report.email], [report.reason], [report.description]   * If any of them is empty, the system shows an error message MSG2 * If [report.phoneNumber] is invalid then the system shows the message MSG30 * If [report.email] is invalid then the system shows the message MSG31 |
| *(9)* | *BR91* | **Message Rules:**  ❖ The system shows an error message MSG2 or MSG30 or MSG31 |
| *(10)* | *BR92* | **Message Rules:**  ❖ The system shows a success message MSG48 |

### 

### UC17: Leave Feedback

| **Name** | **Leave Feedback** |
| --- | --- |
| **Description** | This use case describes the process by which users can leave their feedback |
| **Actor** | Buyer |
| **Trigger** | * When a user clicks the "Feedback" button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The feedback button is clickable and actionable and then redirects the user to the leave feedback form * The user is on the leave feedback form (refer to “Leave Feedback Form” in the “List Description” file) |
| **Post-condition** | * The feedback has been created * The user is redirected to the previous page that they are before clicking on the feedback button |

#### Activities Flow



*Figure 17: Activities Flow of the leave feedback use case*

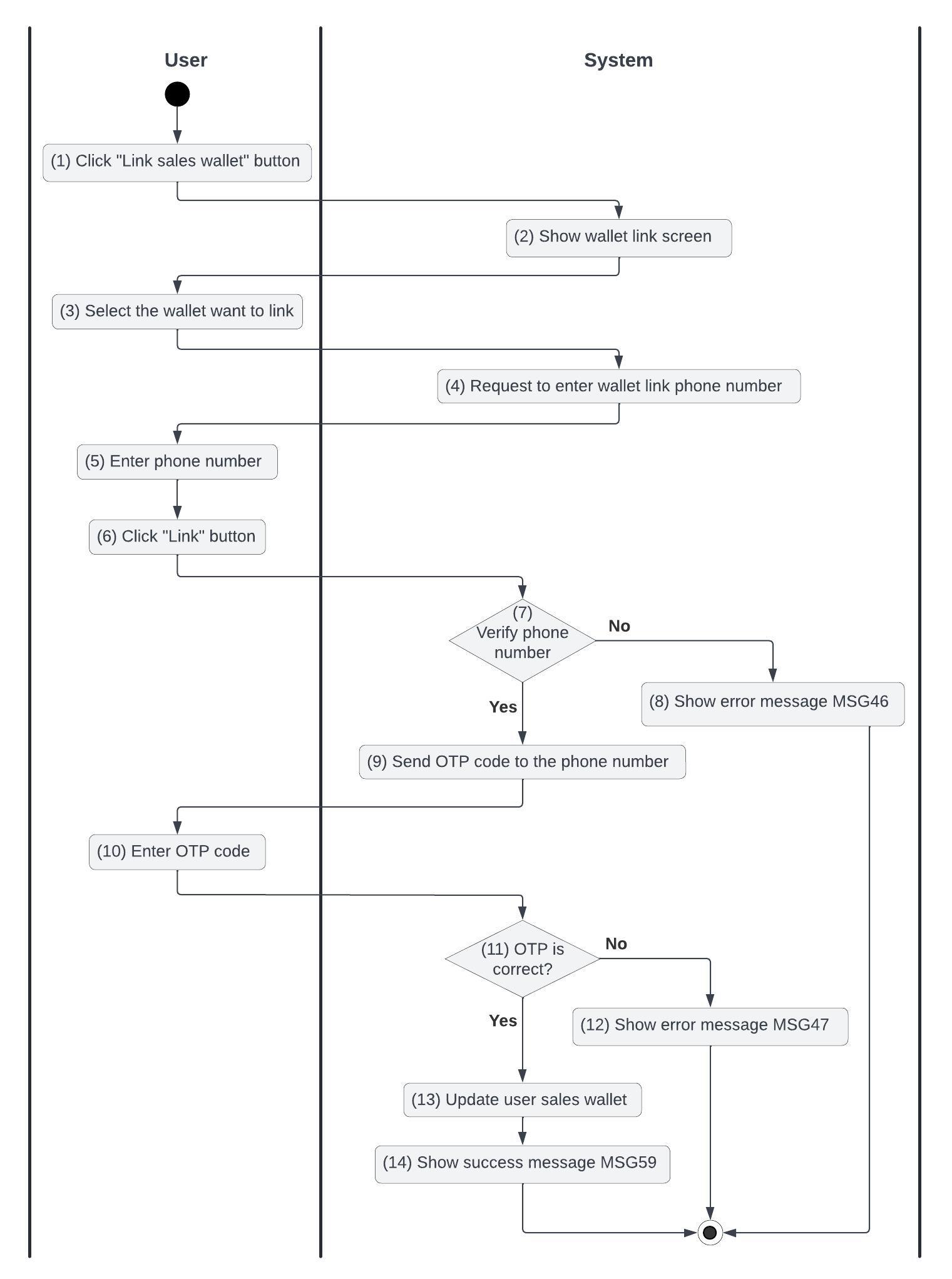
#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR93* | **Direct Rules:**  ❖ The system redirects the user to the leave feedback page (refer to “Leave Feedback Form” in the “List Description” file) |
| *(4)* | *BR94* | **Feedback Form Rules:**  ❖ The system loads the “Leave Feedback Form” (refer to ”Leave Feedback Form” in the “List description” file).  ❖ The form includes the following information fields and the user needs to fill in all fields in this form:   * [feedbackCategory] * [comment] * [proof] ( image/video ) |
| *(6)* | *BR95* | **Creating Rules:**  ❖ When the user clicks the “Save” button, the system will prompt a confirmation message (Refer to MSG1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:   * The system checks the items IsValidFeedbackCategory([feedbackCategory]), IsValidComment([comment]) and IsValidProof([proof]):   + If Isfilled([feedbackCategory]) == False or Isfilled([comment]) == False or Isfilled([proof]) == False is empty, the system shows an error message MSG2. * The feedback will be saved with the items: [feedbackCategory], [comment], [proof] (can be null ), [status] = PENDING, then the system will show the success message MSG57 |
| *(7)* | *BR96* | **Message Rules:**  ❖ The system shows the error message MSG2 |
| *(9)* | *BR97* | **Message Rules:**  ❖ The system shows the success message MSG57 |
| *(10)* | *BR98* | **Redirect Rules:**  ❖ The system redirects the user to the previous page |

### UC18: Link Sales Wallet

| **Name** | **Link Sales Wallet** |
| --- | --- |
| **Description** | This use case describes how a user can link to their merchant wallet |
| **Actor** | Seller |
| **Trigger** | * When a seller clicks the "Link sales wallet" button |
| **Pre-condition** | * The system should be online and accessible * The seller is logging into the website and has internet access * The link sales wallet button is clickable and actionable, then redirects the user to the “Wallet Link’ form * The seller is on the “Wallet Link’ form (refer to “Wallet Link” in the “List Description” file) |
| **Post-condition** | * This seller‘s sales wallet is activated |

#### Activities Flow



*Figure 18: Activities Flow of the link sales wallet use case*

#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR99* | **Loading Screen Rules:**  ❖ The system loads the “Wallet link” screen (refer to “Wallet Link” in the “List Description” file) |
| *(3)* | *BR100* | **Select wallet Rules:**  ❖ The user chooses one of the following wallets:   * Momo * Paypoo * Bank |
| *(5)* | *BR101* | **Entering Rules:**  ❖ The user enters their phone number [phoneNumber] |
| *(7)* | *BR102* | **Phone number verification rules:**  ❖ The system uses the abstract API phone validator to verify phone numbers with the default country code ‘VN’  ❖ If IsValidPhoneNumber([phoneNumber]) is valid then the abstract API return response contains [valid] = true  else the abstract API return response contains [valid] = false and shows the error message MSG46 |
| *(8)* | *BR103* | **Message Rules:**  ❖ The system shows the message MSG46 |
| *(9)* | *BR104* | **Sending OTP Code Rules:**  ❖ The system generates an OTP code to verify the user's phone number with [expire] = 60000 ( mean 1 minute) |
| *(11)* | *BR105* | **Checking OTP Rules:**  ❖ If CheckOTP([OTP\_code]) == True is matched then the system returns a response with status code 200  ❖ else the system returns a response with status code 400 and shows the error message MSG47 |
| *(12)* | *BR106* | **Message Rules:**  ❖ The system shows the error message MSG47 |
| *(13)* | *BR107* | **Update sales wallet rules:**  ❖ If response.status\_code == 200 then  [user.sales\_wallet] = {  “vendor” : <<selected wallet>> (ex: “Momo”),  “phone” : <<encrypted phone number>>  **}** |
| *(14)* | *BR108* | **Message Rules:**  ❖ The system shows the success message MSG59 |

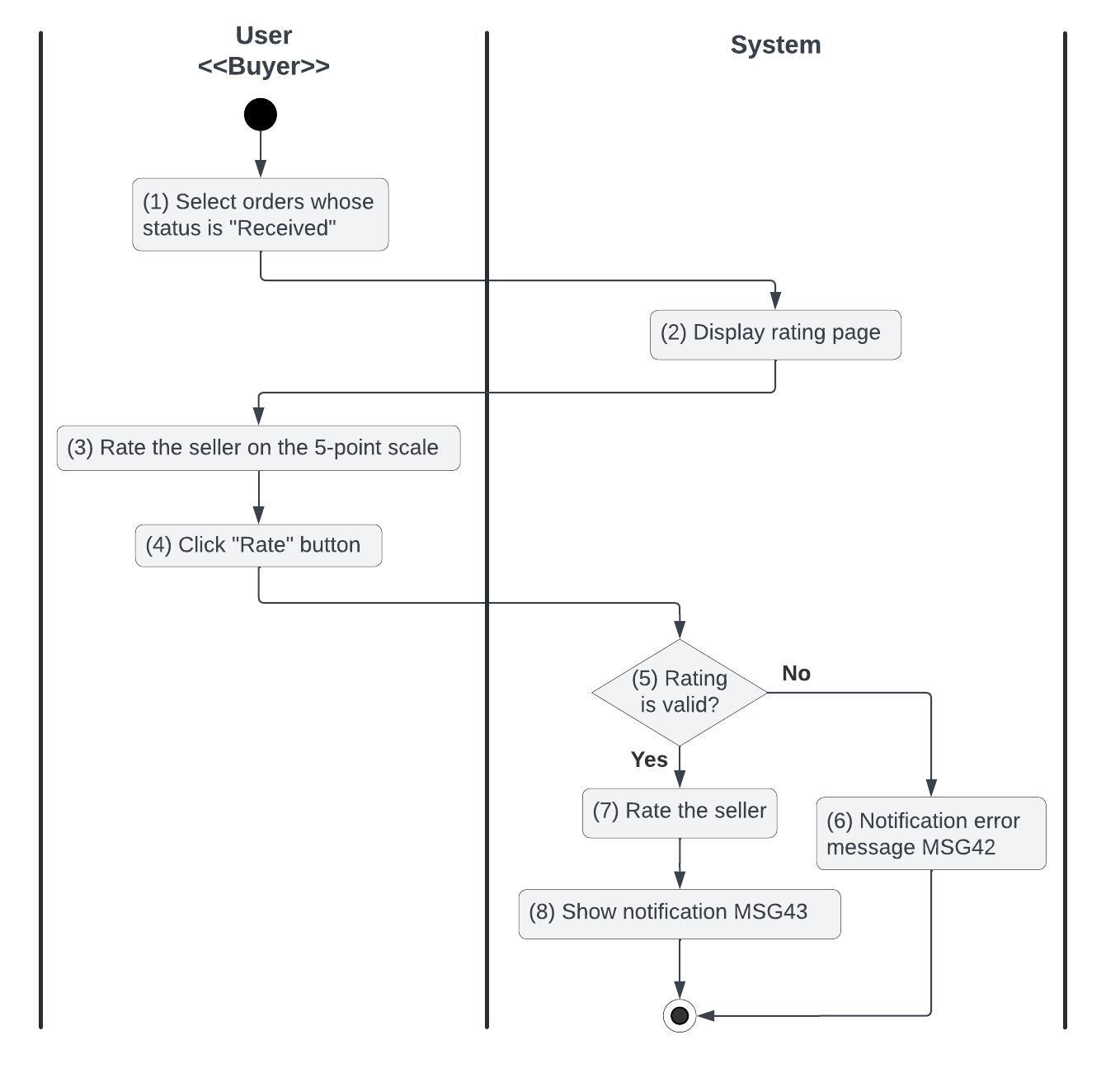
### 

### 

### UC19: Rate Seller

| **Name** | **Rate Seller** |
| --- | --- |
| **Description** | This use case describes how a buyer can rate the seller |
| **Actor** | Buyer |
| **Trigger** | * When the buyer clicks on the “Rate” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The rate button is clickable and actionable * The order has been successfully confirmed to be shipped by the shipping unit * The user is on the Rate page (refer to “Rate Seller Form” in the “List Description” file) |
| **Post-condition** | * Seller’s rating has been saved |

#### Activities Flow



*Figure 19: Activities Flow of the rate seller use case*

#### Business Rules

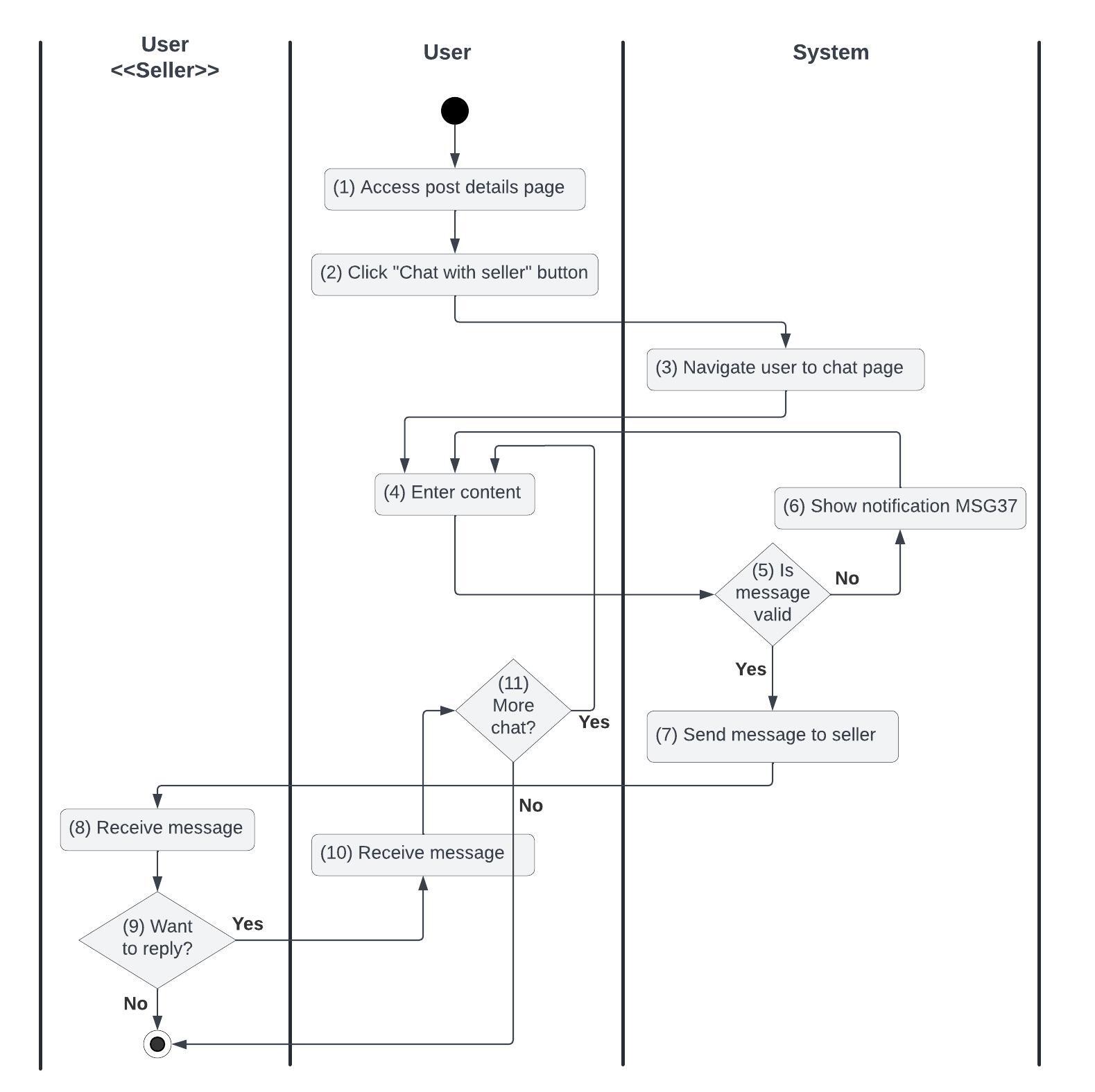
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR109* | **Loading screen rules:**  ❖ The system loads the Rate page (refer to “Rate Seller Form” in the “List description” file) |
| *(3)* | *BR110* | **Rate seller:**  ❖ The user rates the seller by entering a comment [comment] and a rating point [rating] |
| *(5)* | *BR111* | **Validate rules:**  ❖ The system extracts the rating data from requests and checks the items IsValidBuyer[buyerId], IsValidSeller[sellerId], IsValidPost[postId], IsValidComment[comment], and IsValidRating[rating]:   * If IsFilled([comment]) == False or IsFilled([rating]) == False is null or empty then the system returns the error message MSG2. * [buyer] = User repository find by id [buyerId] (call user repository.findById() function) * [seller] = User repository find by id [sellerId] (call user repository.findById() function) * [post] = Post repository find by id [postId] (call post repository.findById() function) * If [buyer] == null then the system returns an error response with status code 400 BAD\_REQUEST * If [seller] == null then the system returns an error response with status code 400 BAD\_REQUEST * If [post] == null then the system returns an error response with status code 400 BAD\_REQUEST |
| *(6)* | *BR112* | **Message Rules:**  ❖ The system shows an error message MSG42 |
| *(7)* | *BR113* | **Rate rules:**   * **[**rate] = Rating repository create new. * [rate.buyer] = [buyer] * [rate.seller] = [seller] * [rate.buyer] = [buyer] * [rate.rating] = [rating] * [rate.comment] = [comment] |
| *(8)* | *BR114* | **Message Rules:**  ❖ The system shows a success message MSG43 |

### 

### UC20: Chat with Seller

| **Name** | **Chat with Seller** |
| --- | --- |
| **Description** | This use case describes how a user can chat with the seller |
| **Actor** | Buyer |
| **Trigger** | * When the user clicks on the “Chat with Seller” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website and has internet access * The chat button is clickable and actionable * The user is on the post details page (refer to “User/Post Details” in the “List description” file) |
| **Post-condition** | * The user is redirected to the Chat page |

#### Activities Flow



*Figure 20: Activities Flow of the chat with seller use case*

#### Business Rules

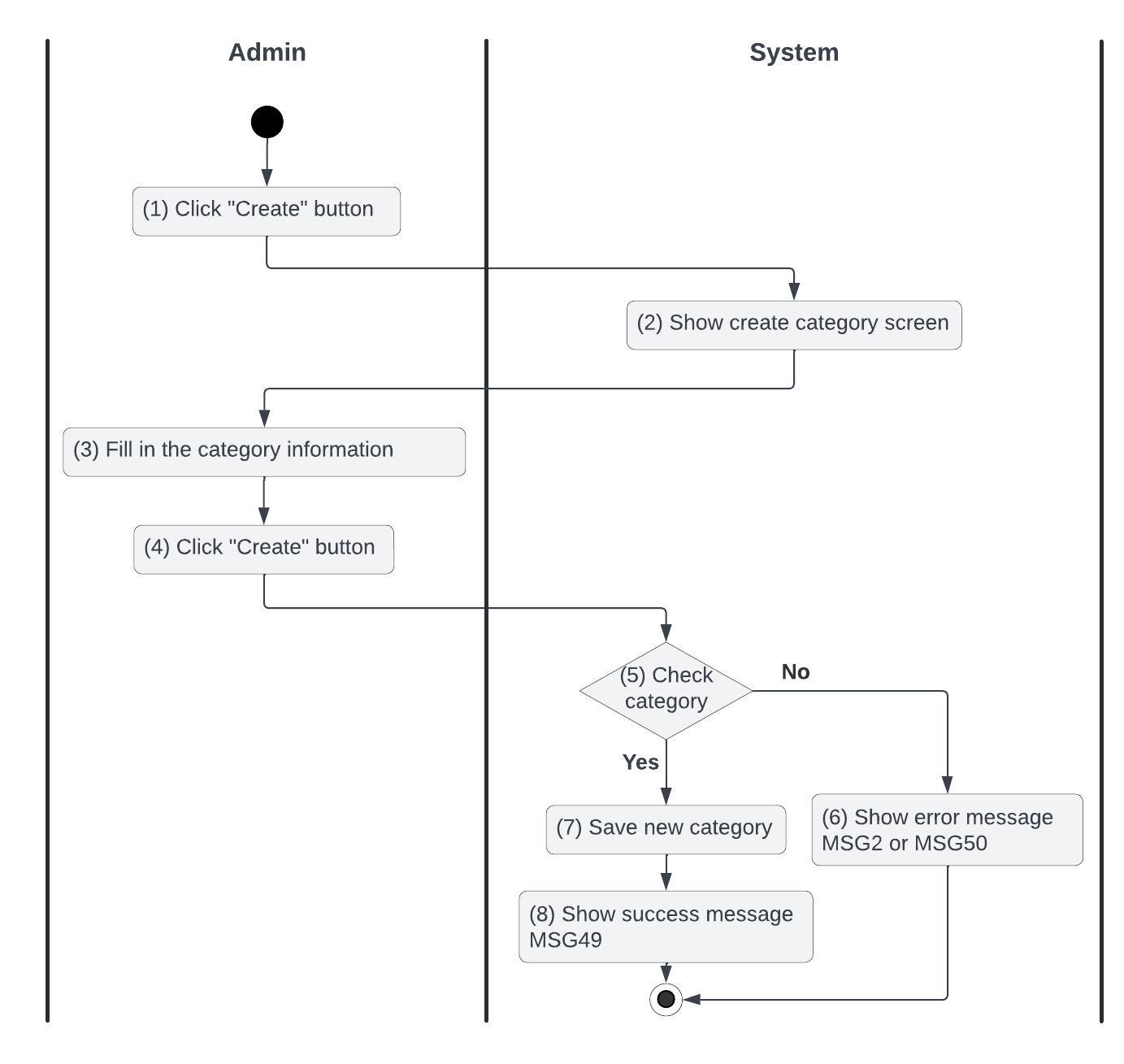
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR115* | **Redirect Rules:**  ❖ The system loads the Chat page (refer to “Chat” in the “List description” file) |
| *(4)* | *BR116* | **Entering Rules:**  ❖ The user enters a message [message] |
| *(5)* | *BR117* | **Validate Rules:**  ❖ The system extracts the [sellerId] the user wants to chat with and [userId] from the request.  ❖ If the socket connection with key [sellerId]-[userId] not exists then [channel] = socket.open()  ❖ The system extracts [message] from the request  ❖ If the [message] contains impolite words then the system shows the error message MSG37   * else [channel].push([message]) |
| *(6)* | *BR118* | **Message Rules:**  ❖ The system shows an error message MSG37 |
| *(9)* | *BR119* | **Reply Rules:**  ❖ The seller enters a message to respond to the user [receivedMessage] |
| *(10)* | *BR120* | **Receive Rules:**  ❖ The user receives a message from the seller and might make a response |

### 

### UC21: Create a Category

| **Name** | **Create a Category** |
| --- | --- |
| **Description** | This use case describes how an admin can create a new category for the system |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Create Category” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The create category button is clickable and actionable * The admin accessed the category screen |
| **Post-condition** | * The new category is created |

#### Activities Flow



*Figure 21: Activities Flow of the create a category use case*

#### Business Rules

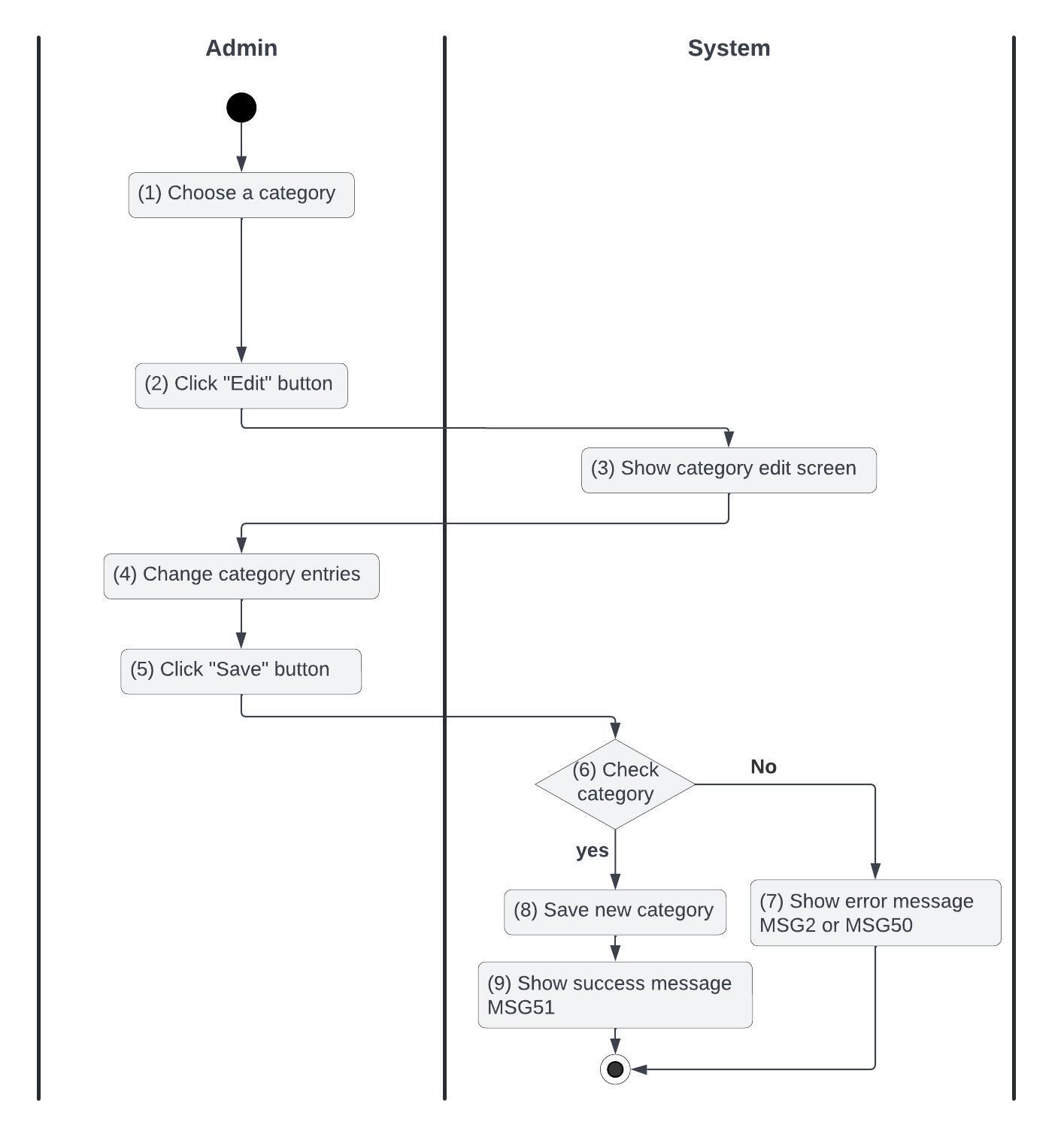
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR121* | **Loading Screen Rules:**  ❖ The system loads the “Create category” screen (refer to “Create Category” in the “List Description” file). |
| *(3)* | *BR122* | **Fill In Category Rules:**  ❖ The system requests the user to fill in the following information:   * [name] * [description] * [brand] = [{   ‘name’:  ‘logo’:  }] |
| *(5)* | *BR123* | **Checking Rules:**  ❖ When the user clicks the “Create” button, the system will prompt a confirmation message (Refer to MSG 1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:  ❖ The system checks the following items: IsValidCategoryName[name], IsValidCategoryDescription[description], IsValidCategoryBrand[brand]   * If IsFilled([name]) == False or IsFilled([description]) == False or IsFilled([brand]) == False is empty, the system shows an error message MSG2. * If DoesCategoryNameExist([name]) == True already exists then the system returns a response with status code 400 and shows an error message MSG50   + else the system returns a response with status code 200 and shows a success message MSG49 |
| *(6)* | *BR124* | **Message Rules:**  ❖ The system shows an error message MSG2 or MSG50 |
| *(10)* | *BR125* | **Message Rules:**  ❖ The system shows a success message MSG49 |

### 

### UC22: Update a Category

| **Name** | **Update a Category** |
| --- | --- |
| **Description** | This use case describes how an admin can change the category information |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Update Category” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The update category button is clickable and actionable * The admin accessed the category screen |
| **Post-condition** | * The category is updated |

#### Activities Flow



*Figure 22: Activities Flow of the update a category use case*

#### Business Rules

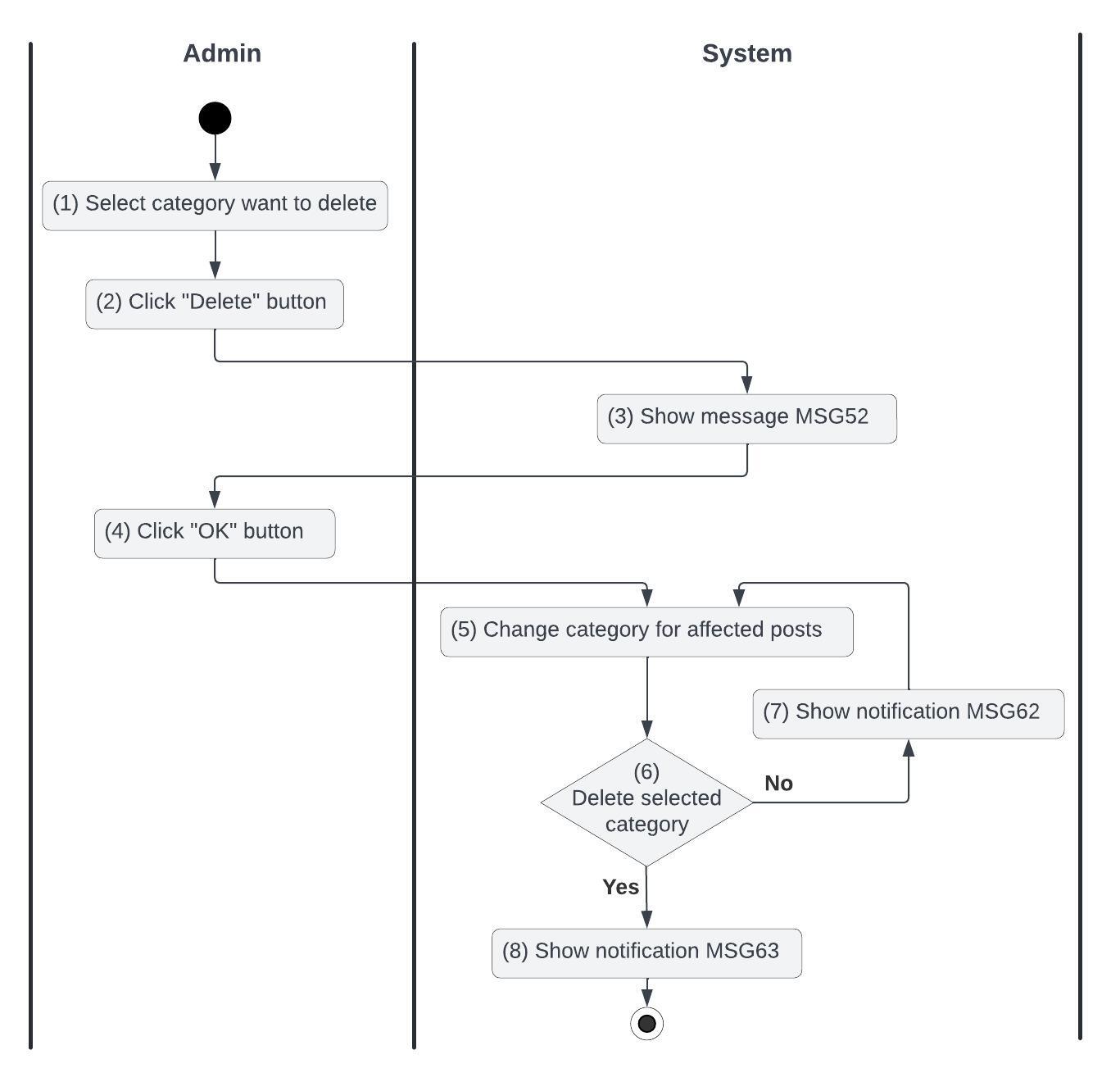
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR126* | **Loading Screen Rules:**  ❖ The system loads the “Update Category” screen (refer to “Update Category” in the “List Description” file). |
| *(6)* | *BR127* | **Checking Category Rules:**  ❖ When the user clicks the “Save” button, the system will prompt a confirmation message (Refer to MSG 1). If the user chooses Cancel, the system does nothing; else, the system will save inputted information and update the item as the following:  ❖ The system checks the following items : IsValidCategoryName[name], IsValidCategoryDescription[description], IsValidCategoryBrand[brand]   * If IsFilled([name]) == False or IsFilled([description]) == False or IsFilled([brand]) == False is empty, the system shows an error message MSG2. * If DoesCategoryNameExist([name]) already exists then the system returns a response with status code 400 and shows an error message MSG50   + else the system returns a response with status code 200 and shows a success message MSG51 |
| *(7)* | *BR128* | **Message Rules:**  ❖ The system shows an error message MSG2 or MSG50 |
| *(9)* | *BR129* | **Message Rules:**  ❖ The system shows a success message MSG51 |

### 

### UC23: Delete a Category

| **Name** | **Delete a Category** |
| --- | --- |
| **Description** | This use case describes how an admin can delete a category |
| **Actor** | Admin |
| **Trigger** | * When the admin clicks on the “Delete Category” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Admin’ permission and has internet access * The delete category button is clickable and actionable * The admin accessed the category screen |
| **Post-condition** | * The categories are deleted |

#### Activities Flow



*Figure 23: Activities Flow of the delete a category use case*

#### Business Rules

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(3)* | *BR130* | **Message Rules:**  ❖ The system shows a message MSG52 |
| *(5)* | *BR131* | **Changing Rules:**  ❖ For each selected category:   * [postRepository].findAllByCategory([selectedCategory]) * for each post found: [post.category] = [categoryRepository].findByName(‘other’) |
| *(6)* | *BR132* | **Deleting Rules:**  ❖ For each selected category:   * If ( [categoryRepository].existById([selectedCategory.id]) == true ) then [categoryRepository].deleteById([selectedCategory]) and then the system shows the success message MSG63   + else the system returns a response with status code 400 and shows message MSG 62 |
| *(7)* | *BR133* | **Message Rules:**  ❖ The system shows an error message MSG62 |
| *(8)* | *BR134* | **Message Rules:**  ❖ The system shows a success message MSG63 |

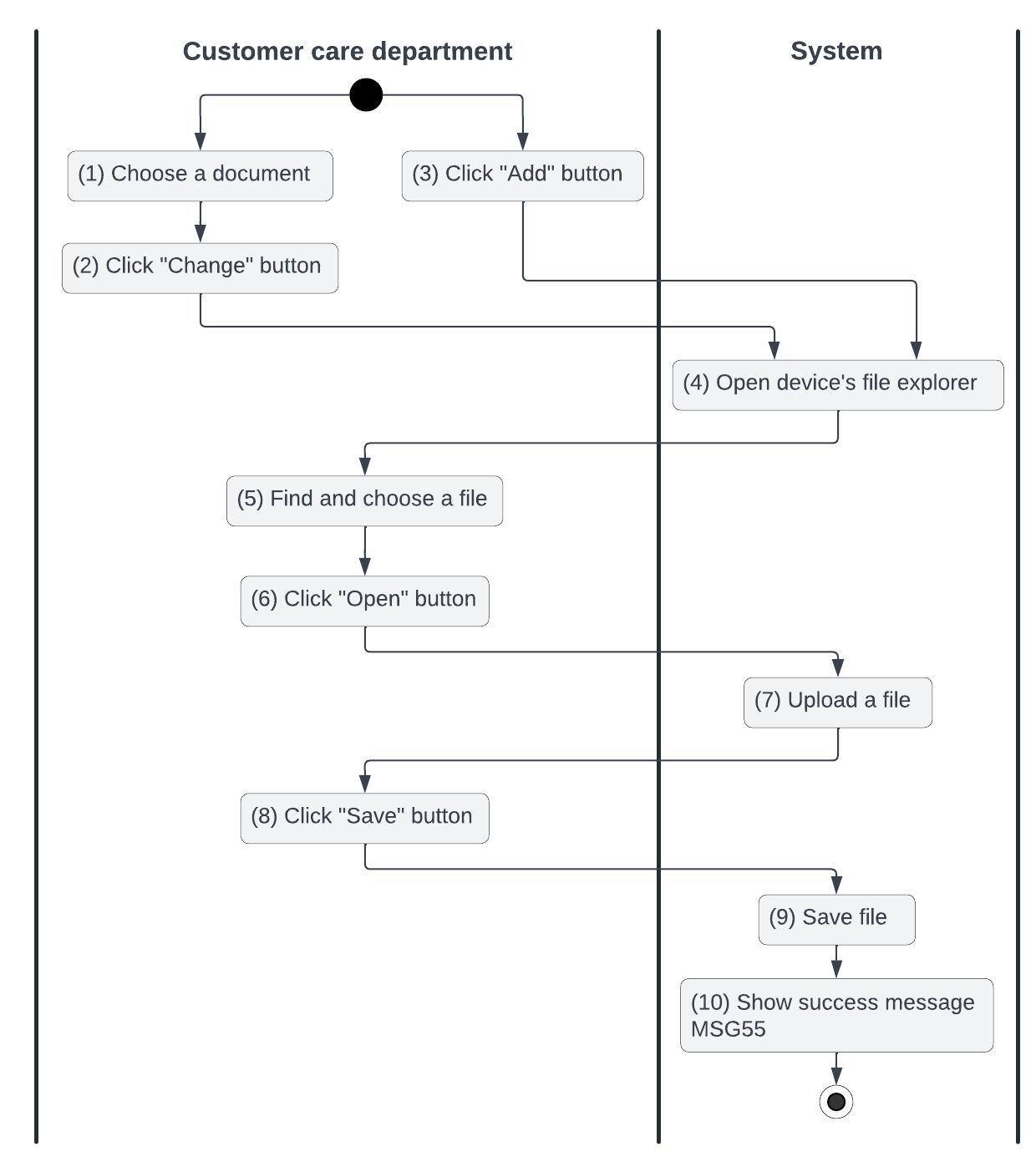
### 

### 

### UC24: Update information for the help center

| **Name** | **Update information for the help center** |
| --- | --- |
| **Description** | This use case describes how users can update content and questions for the help center |
| **Actor** | Customer Care Department |
| **Trigger** | * When the Customer Care Department clicks on the “Update information for the help center” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Customer Care Department’ permission and has internet access * The “Update information for the help center” button is clickable and actionable * The Customer Care Department accessed the document screen |
| **Post-condition** | * The documents are changed |

#### Activities Flow



*Figure 24: Activities Flow of the Update information for the help center use case*

#### Business Rules

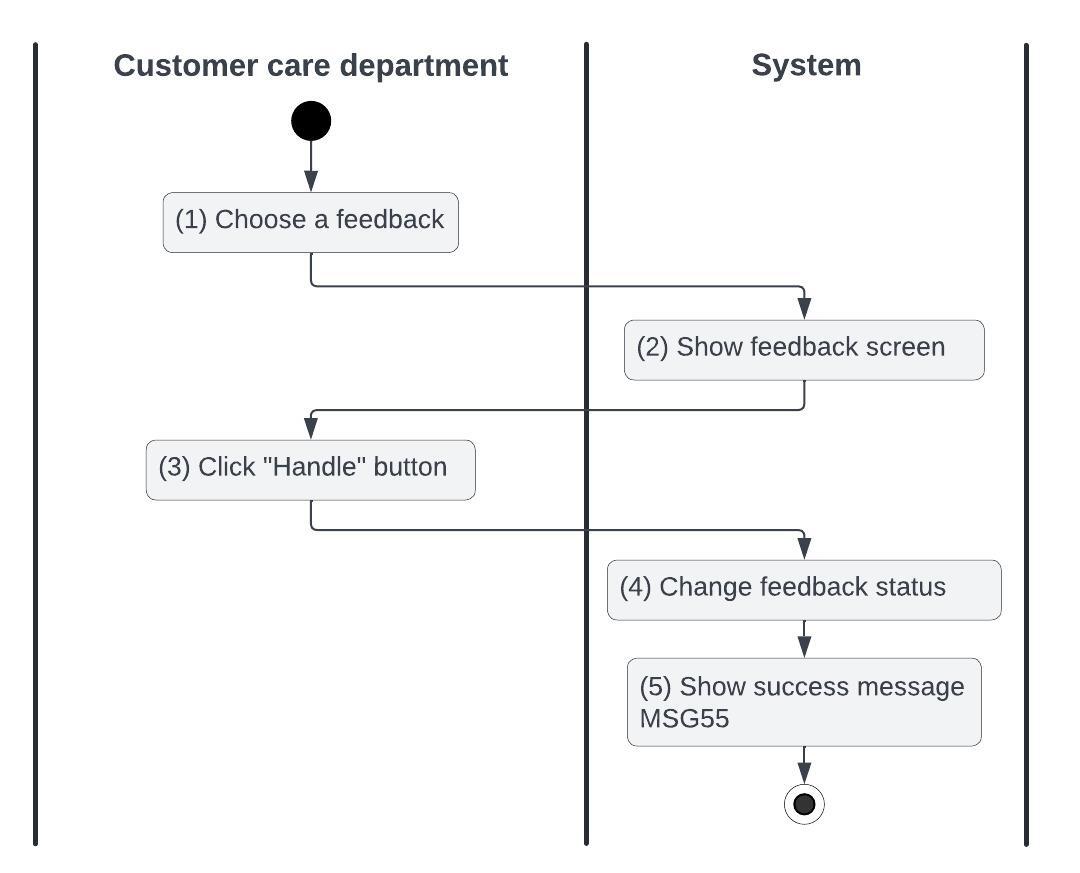
| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(4)* | *BR135* | **Opening File Rules:**  ❖ Require the following file types: .doc, .docx, .pdf |
| *(9)* | *BR136* | **Saving Rules:**  ❖ If the user makes a change then replace the document with the file you just uploaded  ❖ If the user makes an addition then the file you just uploaded will be saved |
| *(9)* | *BR137* | **Message Rules:**  ❖ The system shows a success message MSG55 |

### 

### UC25: Handle feedback from the user

| **Name** | **Handle feedback from the user** |
| --- | --- |
| **Description** | This use case describes how users can view and respond to customer feedback |
| **Actor** | Customer Care Department |
| **Trigger** | * When the Customer Care Department clicks on the “Handle Feedback” button |
| **Pre-condition** | * The system should be online and accessible * The user is logging into the website with ‘Customer Care Department’ permission and has internet access * The “Update information for the help center” button is clickable and actionable * The Customer Care Department accessed the feedback screen |
| **Post-condition** | * The feedback is handled |

#### Activities Flow



*Figure 25: Activities Flow of the handle feedback use case*

#### Business Rule

| **Activity** | **BR Code** | **Description** |
| --- | --- | --- |
| *(2)* | *BR138* | **Loading Screen Rules:**  ❖ The system shows the “Feedback Details” screen |
| *(3)* | *BR139* | **Handling Rules:**  ❖ Users need to carefully consider feedback before clicking the ‘Handle’ button |
| *(4)* | *BR140* | **Changing Rules:**  ❖ [feedback.status] = HANDLED |
| *(5)* | *BR141* | **Message Rules:**  ❖ The system shows message MSG 55 |

### 

# List Description

[](https://docs.google.com/spreadsheets/d/1PiAo-1ArMSA6xvxmSHV9BGrIPYIR-7jJ/edit?gid=75550104#gid=75550104)s

# View Description

[](https://docs.google.com/spreadsheets/d/1YJGL_4B6yjZn_LNt2GIw3B_l3dV0FnhH/edit?gid=404936389#gid=404936389)

# Non-functional Requirements

## User Access and Security

## 

| Actor  Function | Admin | User | Buyer | Seller | Customer Care Department |
| --- | --- | --- | --- | --- | --- |
| Sign In | x | x |  |  |  |
| Sign Up |  | x |  |  |  |
| Forgot password |  | x |  |  |  |
| Change Password |  | x |  |  |  |
| Leave a feedback |  |  | x |  |  |
| View Post |  | x |  |  |  |
| Create post |  | x |  |  |  |
| Filter/Search Post |  | x |  |  |  |
| Save Post to Favorites |  | x |  |  |  |
| Suggest Similar Posts |  | x |  |  |  |
| Update post |  | x |  |  |  |
| Delete post |  | x |  |  |  |
| Link sales wallet |  |  |  | x |  |
| Rate seller |  |  | x |  |  |
| Chat with seller |  |  | x |  |  |
| Report post |  |  | x |  |  |
| Create category | x |  |  |  |  |
| Update category | x |  |  |  |  |
| Delete category | x |  |  |  |  |
| Approve post | x |  |  |  |  |
| Reject post | x |  |  |  |  |
| Ban account | x |  |  |  |  |
| Unban account | x |  |  |  |  |
| Update information for the help center |  |  |  |  | x |
| Handle feedback from the user |  |  |  |  | x |

## 

X: The user has full permission to do the action.

X(\*): The user has permission to do the action on his items.

X(\*\*): The user has permission to do the action on items sent to him only.

## Performance Requirements

**Number of users**

* Number of concurrent users: 300
* Number of business users: 900 - 1000

**Data volume**

* Number of documents: 10M file size
* Data growth rate: 10MB / day

**Level of availability**

❖ 95%: Effective system management (assessed according to IBM standards, continuous operating time per year is no more than 18.25 days).

**Usage frequency**

❖ The system is used regularly, every hour there will be data exchange between businesses and their supply partners. Therefore, the system needs to be set up on a server capable of operating throughout business hours. Upgrades, maintenance, and repairs only take place after hours.

## Implementation Requirements

**Location**

❖ Ho Chi Minh city

**Read-only Duration**

❖ 1 day

**Read-only Timeframe**

❖ 0:00

**Maintenance Window**

❖ Every week on Sunday at 11 p.m., lasting 1 to 2 hours. During this time, programmers can take advantage of it to edit and update new code

**Overall conversion timeline**

❖ 1st and 15th of every month

**Other plans and activities**

[N/A]

# System Requirements

## SharePoint Web Parts

| **#** | **Web Part** | **Description** |
| --- | --- | --- |
| *1* | ContainerWebPart | * To host custom user controls (belong to The Framework source code) |
| *2* | ArchiveEditWebPart | * To implement the archive function (belonging to The Framework source code) |
| *3* | ArchiveListByAgentWebPart | * To implement the archive function (belonging to The Framework source code) |
| *4* | ArchiveListWebPart | * To implement the archive function (belonging to The Framework source code) |
| *5* | AuditSettingsWebPart | * To implement the audit function (belonging to The Framework source code) |
| *6* | AuditViewerWebPart | * To implement the audit function (belonging to The Framework source code) |
| *7* | DialogListWebPart | * To implement the dialog control (belonging to The Framework source code) |
| *8* | PrintPreviewWebPart | * To implement the print function (belonging to The Framework source code) |
| *9* | QuickLaunchContextWebPart | * To implement the quick launch control (belonging to The Framework source code) |
| *10* | QuickLaunchItemWebPart | * To implement the quick launch control (belonging to The Framework source code) |
| *11* | QuickLaunchManagerWebPart | * To implement the quick launch control (belonging to The Framework source code) |

## Custom Pages

There is no custom page implemented in this application.

## Scheduled Agents

There is no planned agent implemented in this application.

## Technical Concern

**Data and user growth**:

* Low growth: Less risk.
* High growth: Scalability challenges.

**Large data volume**:

* Storage and loading issues.
* Slower queries and display times.

**Too much content on a single page**:

* Slower load times.
* Poor user experience.

**Integration with other systems**:

* Performance issues if systems are unstable or slow.

**Database optimization**:

* Necessary for faster queries as data grows.

**Bandwidth and server capacity**:

* Limited bandwidth or weak servers may fail under high traffic.

**Image and video uploads**:

* Large files can increase loading times.

# Appendixes

## Glossary

The list below contains all the necessary terms to interpret the document, including acronyms and abbreviations.

| **Term** | **Description** |
| --- | --- |
| *BR* | **B**usiness **R**ule |
| *CBR* | **C**ommon **B**usiness **R**ule |
| *DB* | Notes **D**ata**b**ase |
| *MSG* | **M**es**s**a**g**e |
| *UC* | **U**se **C**ase |
| *N/A* | **N**ot **A**vailable or **N**ot **A**pplicable, used to indicate when information in a certain section could not be provided because it does not apply to this application. |
| *UI* | **U**ser **I**nterface |
| *SRS* | **S**oftware **R**equirements **S**pecification |
| *TBD* | **T**o **b**e **d**etermined or **t**o **b**e **d**efined |

## Messages

This section describes the details of messages used in business rules e.g. error messages, confirmation messages, etc.

| **Message Code** | **Message Content** | **Button** |
| --- | --- | --- |
| MSG1 | Are you certain about this decision? | OK/Cancel |
| MSG2 | You need to fill in all fields |  |
| MSG3 | Your post is ready to publish. Waiting for the Administrator. |  |
| MSG4 | Payment failed. Please check your account. |  |
| MSG5 | Payment success. |  |
| MSG6 | Are you sure you have received your order? | OK/Cancel |
| MSG7 | Are you certain to approve this post? |  |
| MSG8 | Approve successfully |  |
| MSG9 | Are you certain to reject this post? | OK/Cancel |
| MSG10 | Reject successfully |  |
| MSG11 | The file size is too large |  |
| MSG12 | The product does not exist |  |
| MSG13 | The price must be a number and cannot be less than 0 |  |
| MSG14 | The description must contain at least 50 characters |  |
| MSG15 | You don’t have permission |  |
| MSG16 | Your post has been updated. Waiting for the administrator's approval. |  |
| MSG17 | Updating post failed |  |
| MSG18 | The post does not exist. |  |
| MSG19 | Can not delete the post that has been purchased |  |
| MSG20 | Delete the post successfully |  |
| MSG21 | Delete post failed |  |
| MSG22 | The Username or Password is incorrect |  |
| MSG23 | Signed in successfully |  |
| MSG24 | The username must contain at least 8 characters |  |
| MSG25 | Invalid password |  |
| MSG26 | The phone number has been used. |  |
| MSG27 | Email has been used. |  |
| MSG28 | Successfully registered |  |
| MSG29 | Phone/email has already existed |  |
| MSG30 | Invalid phone number |  |
| MSG31 | Invalid email |  |
| MSG32 | User not found |  |
| MSG33 | Invalid verification link |  |
| MSG34 | The new password must not be the same as the old password |  |
| MSG35 | Changed password successfully |  |
| MSG36 | The post does not exist |  |
| MSG37 | The message contains impolite words |  |
| MSG38 | Invalid payment information |  |
| MSG39 | Payment success. |  |
| MSG40 | Payment failed |  |
| MSG41 | Confirm order successfully. |  |
| MSG42 | Rating buyer failed |  |
| MSG43 | Thanks for your rating. |  |
| MSG44 | This post has been updated with 'Guarantee Payment' |  |
| MSG45 | You need to link to your sales wallet first! | Link/Cancel |
| MSG46 | The phone number is not valid |  |
| MSG47 | OTP is incorrect |  |
| MSG48 | Thank you for your report. We will review and send the verification results to the email you provide |  |
| MSG49 | Your new category has been created |  |
| MSG50 | The category name already exists |  |
| MSG51 | Category update successfully |  |
| MSG52 | ALERT. Do you want to delete these categories? This action can not be undone! | OK/Cancel |
| MSG53 | Account change successfully |  |
| MSG54 | This account is invalid |  |
| MSG55 | The document was updated successfully. |  |
| MSG56 | Handle feedback successfully |  |
| MSG57 | Leave feedback successfully |  |
| MSG58 | Your passwords do not match |  |
| MSG59 | Your sales wall is updated successfully |  |
| MSG60 | You’ve chosen the shipping address successfully |  |
| MSG61 | Your order is canceled successfully |  |
| MSG62 | The category does not exist |  |
| MSG63 | The category is deleted successfully |  |
| MSG64 | Please enter keywords or select at least one filter option |  |
| MSG65 | Price range values must be positive |  |
| MSG66 | The minimum price cannot be greater than the maximum price |  |
| MSG67 | No results were found for your search criteria |  |
| MSG68 | The selected post is no longer available |  |
| MSG69 | This post is already on your favorites list |  |
| MSG70 | The post has been added to your favorites |  |
| MSG71 | No purchase history is available |  |
| MSG72 | No similar posts were found for this item |  |
| MSG73 | Username has already existed |  |
| MSG74 | Your old password is incorrect |  |