If you can implement only one feature, show criteria to prioritize the requirement (5 points)

make user journey from the first feature to be implementing via the link below and List document as follows:

1. URS (of your user journey) 5 points done
2. SRS (of your user journey) 5 points done
3. use case diagram (All features) 2.5 points wait Pattama.. (Beauty) done
4. use case description (of your user journey) 2.5 points Beauty Bay Name done
5. activity diagram (of your user journey) 2.5 points Baibua done
6. Non-functional requirement with fit criteria 2.5 points Name done
7. UI (wireframe or prototype) 5 points Ice done

If you can implement only one feature, show criteria to prioritize the requirement (5 points)

1. Business value: necessary feature that can keep business running.
2. Security: to protect users because insurance companies hold very sensitive data .
3. Users want: features that users want to be included into an app.
4. Cost and effort: cost and effort to complete that feature.
5. Market demand: feature that is hyped in the market to increase market opportunities.
6. URS (of your user journey)

**Claims Submission and Tracking:**

Implement a user-friendly claims submission and tracking system within the app. Travelers should be able to easily submit claims, upload required documentation (e.g., receipts, medical reports), and track the status of their claims in real-time. A streamlined claims process enhances the overall user experience and builds trust with policyholders.



URS-001: Users select a type of coverage they want to claim.

URS-002: Users fill in the details of the claim.

URS-003: Users upload pictures of evidence to claim insurance.

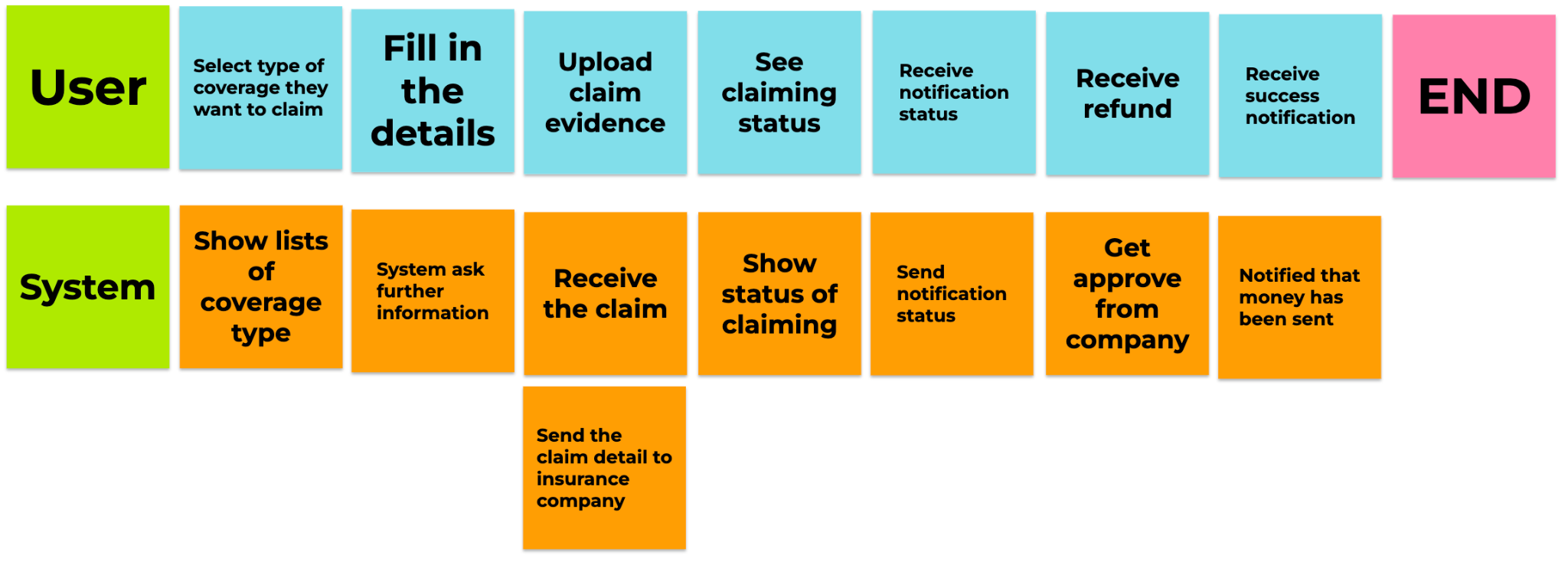
URS-004: Users see details of claiming status.

URS-005: Users receive notification status.

URS-006: Users receive a refund from insurance.

URS-007: Users receive the success refund notification.

1. SRS (of your user journey) (แก้ing)



SRS-001: System shows the lists of coverage types.

SRS-002: System show form for user to fill in the information

SRS-003: System receive the claim from the user

SRS-004: System sends the claim details from the user to the insurance company.

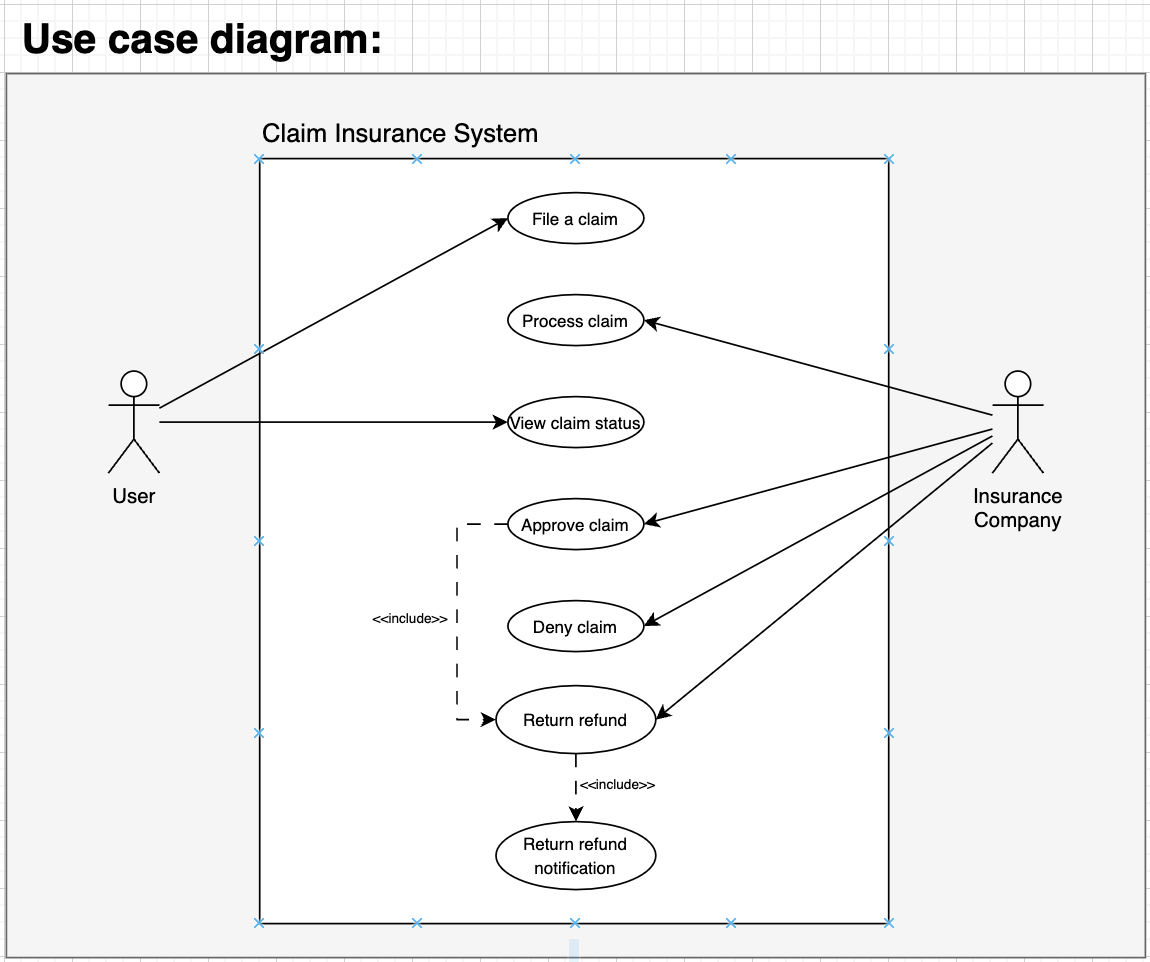
SRS-005: System show status of claiming.

SRS-006: System send notification status.

SRS-007: System receive approval from company

SRS-008: System notified that money has been sent.

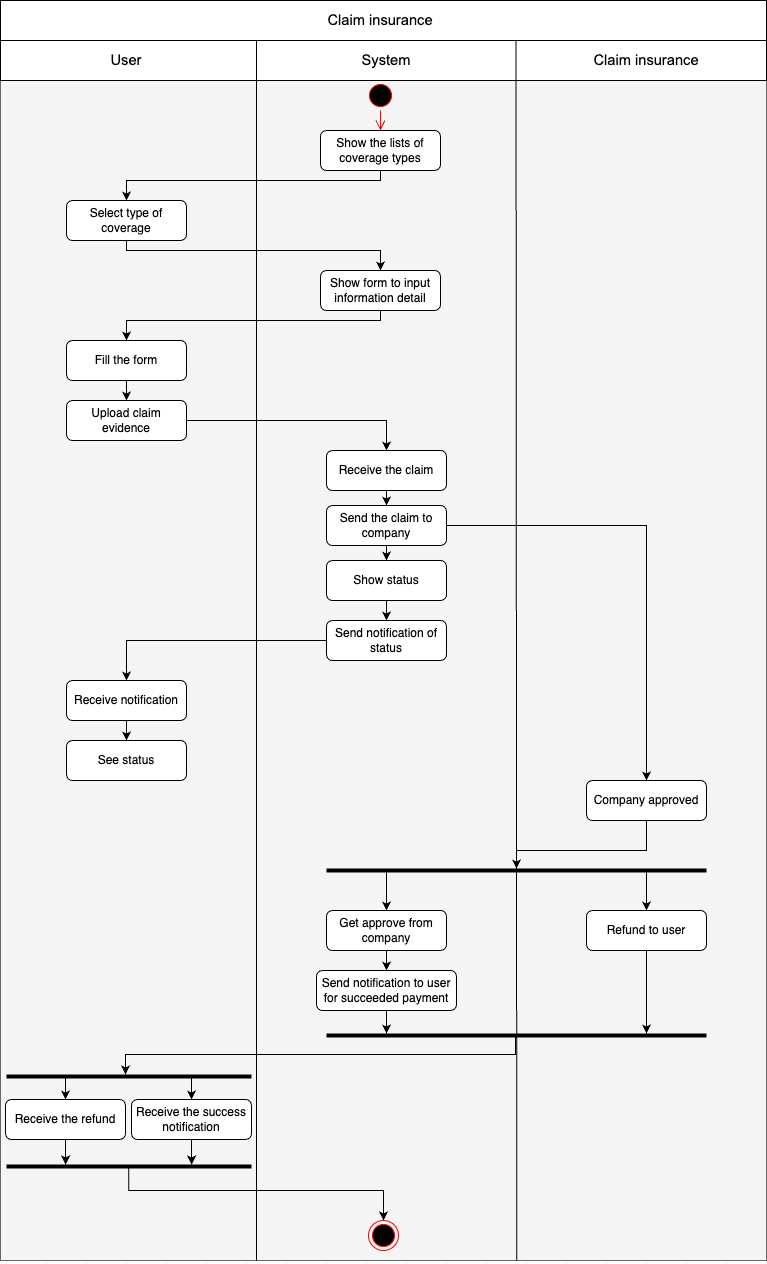
1. Use case diagram



1. Use case description

| Use Case ID | UC-01 | | | | |
| --- | --- | --- | --- | --- | --- |
| Use Case Name | Claim insurance | | | | |
| Created By | Group 1 | | Last Update By | Group 1 | |
| Date Created | 18/09/2023 | | Last Revision Date | 18/09/2023 | |
| Actors | User: customer who want to claim their insurance | | | | |
| Description | User can claim their insurance if their lost is under these coverage   * medical * trip cancellation * lost baggage | | | | |
| Trigger | User click claim button | | | | |
| Preconditions | User has logged in to an app  User has bought an insurance  User has an evidence(s) | | | | |
| Use Case Input Specification | | | | | |
| Input | type | Constraint | | | Example |
| Picture | PNG | image file size must not exceed 0.5 MB | | |  |
| Location | String | Must not be empty | | | CAMT |
| Details | String | Must not be empty | | | Luggage lost at JFK airport |
| Contract | String | Must not be empty | | | 555-234-6677 or fakeEmail@email.com |
| Post conditions | The data must be send to an insurance company | | | | |
| Normal Flows | | | | | |
|  | | User | | System | |
|  | 1. Users press the button to go to the claim page.  3.Users select a type of coverage they want to claim.  5. Users fill in the details of the claim.  6. Users upload pictures of evidence to claim insurance.  10. Users see details of claiming status  12. Users receive notification status.  14. Users receive a refund from insurance  16. Users receive the success refund notification. | | 2. System shows the lists of coverage types  4. System show form for user to fill in the information  7. System receives the claim from the user.  8. System sends the claim details from the user to the insurance company.  9. System shows the status of claiming.[A9.1:The status of the claim is approved]  [A9.2: The status of the claim is not approved]  11. System send notification status.  13. System receive approval from company [E13.1: Cannot connect to the bank account]  15. System notified that money has been sent. | | |
| Alternative Flow | [A9.1 The status of the claim is approved] The system will send notification status. Go to Step 10  [A9.2 The status of the claim is not approved] The system will send notification status “The claim does not meet the conditions.” and cancel that claim. | | | | |
| Exception Flow | [E13.1: Cannot connect to the bank account] Cannot connect to the bank. please do the transaction later. | | | | |
| Assumption | - | | | | |

1. **Activity diagram**



**6. Non-functional requirement**

1. Performance :

=> The system should provide quick response times essential functions, including insurance plan comparison and real-time travel alerts.

=> The system should respond to user actions (e.g., plan comparison, document upload) within 2 seconds under normal load conditions.

=> Transactions related to insurance coverage selection should be completed within 5 seconds.

=> The system should handle a high number of concurrent users without significant degradation in performance.

1. Usability :

=> Users should be able to navigate the app and access important features without requiring a user guide or extensive training.

=> 90% of users should rate the app as "intuitive" or "easy to use" in post-interaction surveys.

=> The app should support at least 5 languages, covering major global regions and demographics.

=> Users should receive updates and important notifications within 24 hours of release.

=> The recommendation system should suggest at least 3 relevant insurance providers based on user preferences.

1. Security :

=> All sensitive user data (e.g., payment information, personal documents) should be encrypted using industry-standard encryption algorithms.

=> The system should pass security audits and penetration tests, identifying no high-severity vulnerabilities.

1. Reliable :

=> The system should maintain 99.9% uptime during standard operating hours.

=> In case of system failure, data recovery should occur within 2 hours with minimal data loss (<1% of critical data).

1. Scalability :

=> The system should handle a 20% increase in user base and data volume without degradation in performance or response time.

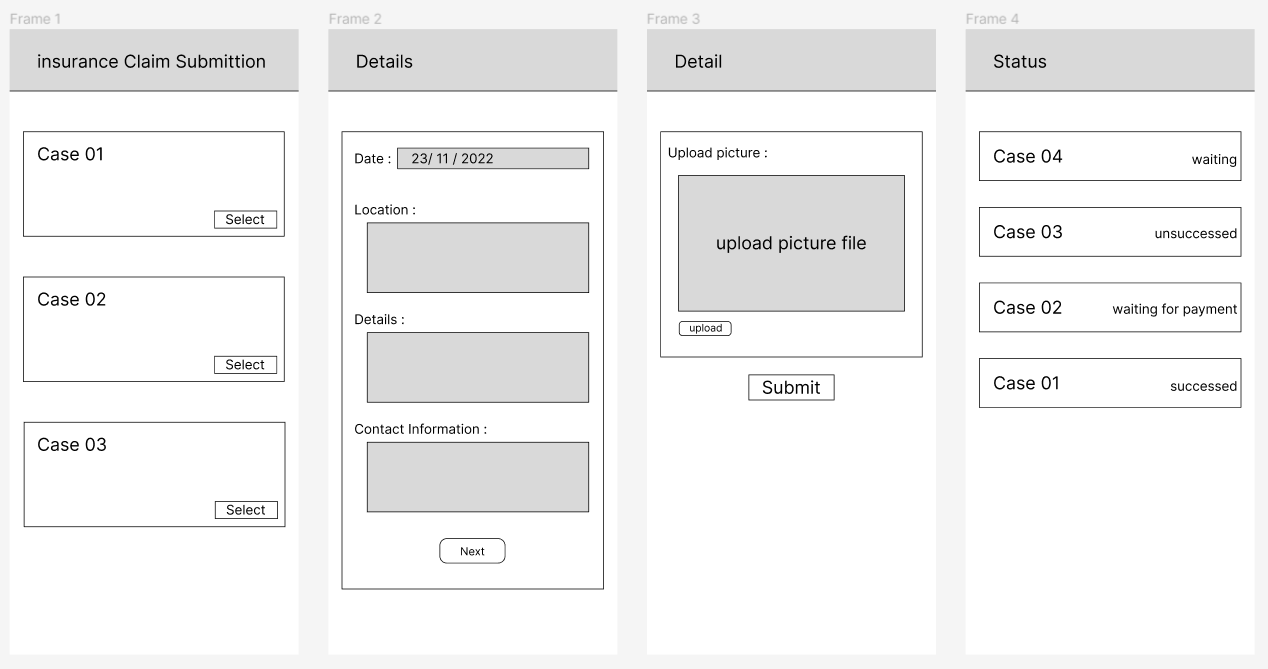
1. Network Resilience :

=> The system should be designed to maintain functionality and usability even in intermittent or low-connectivity network environments

1. **UI ( Wireframe or Prototype )**

Prototype and Wireframe :

<https://www.figma.com/file/4EM1nM9EkyVaAFBGC1kmZC/Untitled?type=design&node-id=3-2&mode=design&t=NRO4t85HCUOBiUTc-0>

Wireframe: 

Prototype: