# **Use Case Document: Post a Squawk**

## **Use Case ID**

UC-001

## **Use Case Name**

Post a Squawk

## **Actors**

* **User**: The individual interacting with the SquawkService to create a new squawk.

## **Preconditions**

* The user is authenticated and logged into the system.
* The user has access to the SquawkService API.

## **Postconditions**

* A new squawk is created and stored in the database.
* The squawk is retrievable by the user and other users.

## **Main Flow**

1. **User Input**: The user provides the content of the squawk (text, images, etc.) through the client interface.
2. **API Call**: The client sends a request to the API endpoint to post a new squawk.
3. **Validation**: The API checks the input for validity (e.g., length, forbidden content).
4. **Command Handling**: The API forwards the request to the appropriate application service.
5. **Domain Logic**: The application service invokes the command handler, which creates a new squawk in the domain model.
6. **Data Persistence**: The command handler saves the new squawk to the NoSQL database via the repository.
7. **Response**: The API returns a success message along with the newly created squawk details (e.g., ID, timestamp).
8. **Update UI**: The client updates the user interface to reflect the new squawk.

## **Alternative Flows**

* **Validation Failure**:
  + If the input is invalid (e.g., too long), the API responds with an error message indicating the validation issues.
* **Database Error**:
  + If there is a problem saving the squawk to the database, the application service catches the exception, and the API returns an error message to the user.
* **Unauthorized Access**:
  + If the user is not authenticated, the API returns an error response prompting the user to log in.

## **Summary**

This use case outlines the basic process of posting a squawk, including the various actors and components involved, as well as the flow of actions that occur in the system. It serves as a foundational use case that can be expanded upon to include additional functionalities and requirements for the **SquawkService**.