**User Story: Feedback History & Tracking**

*“As a feedback provider, I want to track the status of my feedback, so I can understand where it's at in the review process and what actions are being taken.”*

**Story Tasks**

1. **Research and Requirement Gathering**

* Conduct interviews or surveys with feedback providers to understand their needs related to feedback tracking.
* Analyze current feedback mechanisms in the system to identify the absence of tracking features.

1. **Design**

* Design a user interface that allows feedback providers to view a list of their submitted feedback.
* Include filters or search features to navigate through the feedback list easily.
* Design a detailed view for each feedback entry showing its current status and any associated actions.

1. **Backend Development**

* Create or modify database structures to store feedback status and actions.
* Implement API endpoints to retrieve feedback history for a particular user and update feedback status.

1. **Frontend Development**

* Implement the feedback history UI based on the approved design.
* Integrate the UI with backend API endpoints to fetch and display feedback data.

1. **Testing**

* Conduct unit tests for backend functionalities.
* Perform integration tests to ensure frontend and backend work seamlessly.
* Organize user acceptance testing sessions with feedback providers.

1. **Documentation**

* Update user manuals or guides to include the feedback history and tracking feature.

1. **Deployment**

* Plan and schedule the deployment of the new feature.
* Monitor post-deployment to ensure stability and functionality.

**Functional Requirements**

1. **Feedback History UI**

* The system shall provide a user interface where feedback providers can view a list of their submitted feedback.
* Feedback entries should display a summary, date of submission, current status, and any associated actions or comments.

1. **Feedback Status**

* Each feedback entry shall have an associated status (e.g., Submitted, Under Review, Addressed, Closed).
* Feedback providers should be able to click on an entry to view more detailed information and history.

1. **Filters and Search**

* Feedback providers shall have the option to filter their feedback based on status, date, or other relevant criteria.
* A search feature should allow feedback providers to quickly find specific feedback entries.

**Backend Integration**

* The system shall store feedback history, status updates, and associated actions in a structured manner.
* Feedback providers shall be able to fetch their feedback history through authenticated API calls.

**Non-functional Requirements**

1. **Usability**

* The feedback history interface should be intuitive and user-friendly, ensuring easy navigation and understanding.

1. **Performance**

* The system should quickly load feedback history without excessive delays.

1. **Security**

* Feedback history should be accessible only to the respective feedback provider, ensuring user data privacy.
* The system should implement strong authentication and authorization mechanisms.

1. **Scalability**

* As the number of feedback submissions grows, the system should efficiently handle an increasing volume of data without degrading performance.

1. **Reliability**

* Feedback history and tracking data should be consistently available without frequent downtimes or errors.

1. **Maintainability**

* The feedback history and tracking feature should be implemented in a modular manner to allow easy updates or modifications in the future.

1. **Accessibility**

* Ensure the feedback history user interface complies with accessibility standards, making it usable by all, including those with disabilities.