360-Degree Feedback Application

For this case study let us consider a peer feedback application where a given user in an organization (recipient) can request feedback from other users (submitters) in the same organization.

High-level requirements for this application consist of the following:

- The recipient can identify 5-10 submitters in their organization to request feedback to be given about the recipient.
- Feedback entered by the submitter consists of answering a number of questions about the recipient..
- Each organization may create their own questions or choose questions from a question bank pre-populated in the application.
- The submitter has the option for their feedback to be anonymous.
- The date of the feedback submission should be recorded and displayed to the recipient.
- The recipient can view the contents of the feedback – including date, feedback, and submitter (if not anonymous).
- The recipient can view a list of all feedback submitted to them, including average score across all submitters
- The recipient could request the above to happen multiple times (for example: in the beginning of the year and in the middle of the year) and average score need to be shown per time period and not across all answers

Given the above requirements we ask that you construct the **UI** for all roles (e.g. submitter & recipient) as well as all **data models** necessary to support this application. Deliverables should consist of:

1. **Wireframes** that demonstrate the application flow. These wireframes can be created in a tool of your choice (e.g. Omnigraffle, Sketch, etc.) or clearly hand drawn and scanned.
2. Data Model(s) in the form of **complete physical ERD normalized diagrams** that support your feature functionality. This should include a comprehensive series of entity relationship diagrams covering **all** entities required to support the application you described in #1 above. This information will be used to gauge your domain modeling abilities – so be as detailed as you can. Example items to include per entity include main attributes of each entity, foreign keys, etc.).
3. A **database query** based on the above data model design, that if given a particular Survey ID would return the question text, submitter response, and the name of the submitter.

As you work through your response don't feel limited to only the above use cases. **Feel free to expand on the above requirements to include any other features or functionality you believe will enhance the value of the application.** This being said focus on aspects core to the above uses cases. Solutions outside the core scope including account setup, password retrieval, login, etc. are not necessary for this case study.

Regarding deliverable format we encourage you to use whatever tools best express your solution. Submissions in the form of Microsoft Word documents, PDF files, hosted wireframes, and online clickable walkthroughs are all acceptable. Where possible try and avoid complex archives (e.g. ZIP containers) as they can be difficult to properly interpret.

Lastly please note any assumptions used in your solution that will help us better understand your design and intent.

Algorithm:

In line with the below requirement, write an algorithm to display an “average score across all submitters per feedback request.”

The recipient can view a list of all feedback submitted to him or her, including an average score across all submitters and all feedback.

You may use pseudocode to elaborate on your logic through the function/method.

Your function should take all relevant pieces of input, process information and return an output score.

Please specify your assumptions used to make the score normalized and comparable.

You may assume the existence of any functions required for database/backend connections to retrieve data.