**1. Provide an example of five hypothetical non-functional requirements for this system. Be sure to include the specific type of requirement discussed in class, with each requirement coming from a unique category**.

1. **Usability:** the system should be intuitive and user friendly; new users should not need more than 5 minutes to understand how to use the system
2. **Performance:** the system should load and display tasks within 2 seconds
3. **Reliability:** the system should be available 99% of the time with scheduled maintenances
4. **Supportability:** the system should be usable in multiple languages
5. **Implementation/Constraints:** system must be written in Java or Python

**2. Provide an example of five hypothetical functional requirements for this system.**

1. The system must allow users to create, edit, and delete tasks with titles, descriptions, due dates, and priority levels.
2. Users can sort tasks by priority, due date, or status.
3. Users must login to access their tasks, with password recovery options
4. Users can set reminders for tasks and receive notifications
5. Users can share tasks and collaborate with others on shared tasks

**3. Think of a specific task required to complete each of the functional requirements and non-functional requirements mentioned above (10 total). Estimate the amount of effort needed to complete this task using function points (i.e., using the values** [**here**](https://www.scrumpoker-online.org/en/room/44441241/scrum-poker)**). Briefly explain your answer.**

1. Conduct usability testing and refine the user interface based on results
   1. 13 points. Designing usability tests and recruiting participants is time-consuming and possibly not fully accurate.
2. Optimize database queries and frontend rendering.
   1. 8 points. Optimizing these things would allow for faster loading times, but can be difficult to implement.
3. Test for system reliability and schedule maintenances
   1. 5 points. Stress testing the system shouldn’t be too challenging, but it can be time consuming and planning for maintenance allows us to properly maintain uptime
4. Implement multi-language support
   1. 3 points. This allows people that don’t speak English to use the system, but our translations might not be accurate and some text encoding doesn’t work with non Latin based languages.
5. Choose and implement system in specific language
   1. 13 points. This technically includes creating the entire system, but choosing a language on its own is not difficult.
6. Develop test creation, editing, and deletion features
   1. 3 points. These are simple function in isolation, but incorporating them within the system may prove to be challenging
7. Implement task sorting functionality
   1. 1 point. This is not difficult and just requires our task system to be implemented beforehand
8. Develop user authentication and password recovery features
   1. 3 ponts. Having done such features before this is not too challenging, but it can be difficult to protect user data
9. Develop reminder system with appropriate notifications
   1. 3 points. This is not challenging and should be trivial to implement
10. Develop task sharing and collaboration features
    1. 5 points. This encompasses the user account system and will require the ability to send data from one user to another

**4. Write three user stories from the perspective of at least two different actors. Provide the acceptance criteria for these stories.**

Actor 1: regular user

* As a regular user, I want to be able to create a new task so that I can organize my to-do list.
  + Acceptance criteria: I should be able to add a task with a title, description, and priority.
* As a regular user, I want to receive reminders for my upcoming tasks to stay on top of my schedule.
  + Acceptance criteria: I should receive a notification at a set time before the task’s due date.

Actor 2: Collaborator

* As a collaborator, I want to be able to share tasks with team members and edit them together for effective teamwork.
  + Acceptance criteria: Team members should be able to access and edit tasks shared with them in real-time.

**5. Provide two examples of risk that could potentially impact this project. Explain how you would mitigate these risks if you were implementing your project as a software system.**

* **Data breach risk:** The risk of a data breach could cause user trust to fall and could create legal issues as well.
  + **Mitigation:** Use effective security measures to ensure data encryption.
* **Technical complexity risk:** The project may include more complexity than initially planned, which could result in an underperforming or unfinished product.
  + **Mitigation:** Maintain a well-defined scope, and conduct regular feasibility testing.

**6. Describe which process your team would use for requirements elicitation from clients or customers, and explain why.**

Surveying would be our main method of requirement elicitation. This app is wholly intended to be useful to clients and customers, so not finding out what specific features they want would render our product useless. A to-do app that no one likes the format of entirely fails as an effective to-do app.