**Lesson: Gathering Feedback to Inform Future Development**

**Learning Objectives:** By the end of this lesson, you will be able to:

* Understand the importance of gathering feedback in the development process.
* Use appropriate tools, methods, and techniques to demonstrate a prototype to different audiences.
* Develop a structured plan for gathering feedback.
* Record and organise feedback for analysis and future improvements.

**1. Why is Feedback Important?**

Feedback helps developers:

* Identify strengths and weaknesses in their prototype.
* Understand user needs and expectations.
* Improve functionality, usability, and overall user experience.
* Plan future development more effectively.

**2. Preparing to Gather Feedback**

**A. Demonstrating the Prototype**

Your prototype must be demonstrated to two types of audiences:

1. **Technical Audience (e.g., Programming Professionals):**
   * Focus on technical aspects like code efficiency, scalability, security, and integration.
   * Demonstrate architecture, algorithms, and performance benchmarks.
   * Tools: Code documentation, technical presentations, debugging tools, performance metrics.
2. **Non-Technical Audience (e.g., Client, End-Users):**
   * Focus on usability, accessibility, and visual appeal.
   * Explain key features in simple terms.
   * Demonstrate how the prototype meets user needs.
   * Tools: User manuals, walkthrough videos, surveys, guided testing sessions.

**B. Planning for Feedback Collection**

A structured approach ensures useful feedback. Your plan should include:

1. **Objectives:** Define what you aim to learn from the feedback.
2. **Target Audience:** Identify the groups providing feedback.
3. **Methods:** Choose suitable tools and techniques.
4. **Questions:** Prepare clear and concise questions to gather relevant insights.
5. **Format:** Decide how to record and store the feedback.

**3. Gathering Feedback Methods**

| **Method** | **Technical Audience** | **Non-Technical Audience** |
| --- | --- | --- |
| **Surveys** | Questions about performance, security, and efficiency. | Questions about ease of use, design, and accessibility. |
| **Interviews** | In-depth discussion on code structure and system integration. | User experience, difficulties faced, and suggestions. |
| **Usability Testing** | Developers test functionality with edge cases. | Users perform tasks to evaluate ease of use. |
| **Bug Reports** | Identify technical issues and potential fixes. | Identify usability or navigation issues. |

**4. Recording and Storing Feedback**

Feedback should be structured for easy analysis. Consider using:

* **Spreadsheets:** Categorise responses by type and severity.
* **Documents:** Store qualitative responses.
* **Video/Audio Recordings:** For reference and review.
* **Issue Tracking Tools:** Log technical problems (e.g., Jira, Trello).

**File Naming Convention:**

All materials should be saved using the following format:

Task3\_PartA\_[Document name]\_[Registration number]\_[Surname]\_[First letter of First name]

Example:

Task3\_PartA\_FeedbackSurvey\_123456\_Smith\_J

**5. Next Steps: Using Feedback to Inform Development**

Once feedback is collected:

1. **Analyse:** Identify common issues or suggestions.
2. **Prioritise:** Rank issues based on impact and feasibility.
3. **Plan Improvements:** Define changes for the next iteration.
4. **Document Changes:** Keep a record of updates for future reference.

**Conclusion:** Gathering and analysing feedback is essential for refining your prototype. By engaging both technical and non-technical audiences, you ensure that your product meets both performance and usability expectations.

**Assignment:**

1. Develop a demonstration plan for both audiences.
2. Create feedback collection materials (e.g., surveys, interview questions).
3. Gather and record feedback from test users.
4. Submit all materials following the naming convention.