**A close-up of a text

Description automatically generated**

**Q.1.1 What are the four project categories delineated by the matrix combining goal and solution clarity?**

**Answer:** The four project categories are:

1. **Clear Goal and Clear Solution**: Engineering Projects
2. **Clear Goal but Unclear Solution**: Research and Development Projects
3. **Unclear Goal but Clear Solution**: Exploratory Projects
4. **Unclear Goal and Unclear Solution**: Discovery or Innovation Projects

**Q.1.2 Propose suitable project life cycle methodologies for each project category and provide a rationale for your selections.**

**Answer:**

1. **Engineering Projects (Clear Goal and Clear Solution)**:
   * **Methodology**: Waterfall
   * **Rationale**: Since both the goal and solution are clear, a linear and structured Waterfall approach works best for predictable, sequential tasks.
2. **Research and Development Projects (Clear Goal but Unclear Solution)**:
   * **Methodology**: Iterative or Agile
   * **Rationale**: The goal is clear but the solution evolves. Agile allows for frequent adjustments while maintaining a focus on the end goal.
3. **Exploratory Projects (Unclear Goal but Clear Solution)**:
   * **Methodology**: Lean
   * **Rationale**: The goal is unclear, and Lean allows for continuous refinement of goals while working with a clear solution.
4. **Discovery or Innovation Projects (Unclear Goal and Unclear Solution)**:
   * **Methodology**: Scrum or Design Thinking
   * **Rationale**: Scrum or Design Thinking offers iterative and adaptive processes to refine both goals and solutions.

**A close-up of a document

Description automatically generated**

**Q.2 What sequence of supported functions will you prioritize across the three tiers in your development process for the three-tier web application?**

**Answer:**

1. **First Set**: Develop the **Create Order** function, as it's fundamental to initiating the workflow.
   * Implement across Tier-1 (user interface), Tier-2 (application logic), and Tier-3 (database insertion).
2. **Second Set**: Develop the **Display Orders** function so users can view existing orders.
   * Implement across Tier-1 (display orders), Tier-2 (retrieve orders logic), and Tier-3 (fetching orders from the database).
3. **Third Set**: Develop the **Update Orders** and **Cancel Orders** functions to manage existing orders.
   * Implement across all three tiers for order modifications and deletions.

**A close up of text

Description automatically generated**

**A black text on a white background

Description automatically generated**

**Q.3.1 Create a rough illustration of a Project Life Cycle suitability evaluation chart based on the provided attributes and scores.**

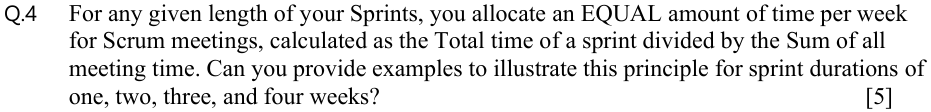
**Answer:**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Q.3.2 Recommend a project life cycle model tailored to this project’s requirements. Justify your choice based on the evaluation scores and project characteristics.**

**Answer**:

* **Project Life Cycle**: Agile
* **Rationale**: The low scores in decision-making, delivery, and change suggest constant requirement evolution. Agile provides flexibility to accommodate frequent changes while delivering working increments. The high criticality score ensures the need for constant feedback and progress monitoring.

****

**Q.4 Can you provide examples to illustrate the principle of allocating equal time per week for Scrum meetings for sprint durations of one, two, three, and four weeks?**

**Answer**:

1. **1-week Sprint**:
   * Total time: 5 days
   * Meeting time: 1 hour daily × 5 days = 5 hours
   * Weekly time allocation: 5 hours
2. **2-week Sprint**:
   * Total time: 10 days
   * Meeting time: 1 hour daily × 10 days = 10 hours
   * Weekly time allocation: 5 hours per week
3. **3-week Sprint**:
   * Total time: 15 days
   * Meeting time: 1 hour daily × 15 days = 15 hours
   * Weekly time allocation: 5 hours per week
4. **4-week Sprint**:
   * Total time: 20 days
   * Meeting time: 1 hour daily × 20 days = 20 hours
   * Weekly time allocation: 5 hours per week

**A close-up of black text

Description automatically generated**

**Q.5.1 Create three child user stories from the following Epic user story: "As a traveler, I want to be able to save favorites on my mobile weather app so I can choose from a finite drop-down list to easily locate the weather in the destination I am traveling to."**

**Answer**:

1. As a traveler, I want to add a location to my favorites list on the mobile weather app so I can quickly access it later.
2. As a traveler, I want to delete a location from my favorites list so that I can manage my preferred destinations.
3. As a traveler, I want to view my saved favorite locations in a drop-down list so I can select a destination for weather updates.

**Q.5.2 Choose one of the child user stories that you just created and write two acceptance criteria for it.**

**Answer** (For user story 1):

1. The system must allow users to save a location in the favorites list when selecting a location from the search results.
2. The system must display a success message confirming the location has been added to the favorites list.

**A white background with black text

Description automatically generated**

**Q.6 List and explain briefly at least two commonly used XP work practices in each of the following categories:**

**Answer**:

1. **Requirements Practices**:
   * **User Stories**: Requirements are captured in simple user stories that describe functionality.
   * **Acceptance Tests**: Each user story has defined acceptance criteria to verify its completion.
2. **Team/Organization Practices**:
   * **Collective Code Ownership**: Any team member can modify any part of the code, promoting shared responsibility.
   * **Pair Programming**: Two developers work together on the same code, ensuring continuous review and higher quality.
3. **Software Development Practices**:
   * **Test-Driven Development (TDD)**: Writing tests before writing the code to ensure all functionality is tested.
   * **Refactoring**: Continuously improving the code without changing its external behavior.
4. **Integration Testing and Release Practices**:
   * **Continuous Integration**: Frequently merging code to detect integration issues early.
   * **Frequent Releases**: Releasing working software increments regularly for feedback and improvements.

**A close up of a text

Description automatically generated**

**Q.7.1 a) Calculate the average story point estimate for the completed user stories.**

b) How would you determine the most appropriate story point estimate for a new user story estimated between 5 and 8 points?

**Answer**:

a) **Average Story Point Estimate**:

* Completed stories: 3, 5, 8, 13, 21
* Average = (3 + 5 + 8 + 13 + 21) / 5 = 50 / 5 = **10**

b) **Most Appropriate Story Point Estimate**:

* The new user story is estimated between 5 and 8 points.
* Using the Fibonacci sequence and team history, the estimate should lean towards **8 points**, since Agile estimation generally rounds up when uncertainty or complexity is higher.

**Q.7.2 Define the core principles and practices of Kanban within the context of software development. Explain how Kanban differs from traditional Agile methodologies such as Scrum.**

**Answer**:

1. **Core Principles of Kanban**:
   * **Visualize Workflow**: Use a Kanban board to track task progress across stages.
   * **Limit Work in Progress (WIP)**: Set limits on the number of ongoing tasks to avoid overloading the team.
2. **Differences from Scrum**:
   * **Kanban** allows continuous delivery without fixed sprints, whereas **Scrum** works in fixed sprints.
   * **Kanban** focuses on workflow optimization, while **Scrum** emphasizes timeboxed iterations and roles.
3. **Continuous Delivery**:
   * Kanban promotes constant flow by visualizing tasks and limiting WIP, leading to continuous delivery of work into production rather than waiting for the end of a sprint as in Scrum.