**Project Management Guide**

**Objective:**

This guide helps each team member know how to structure, write, and submit their assigned sections of the PMP document. Please follow the instructions carefully to ensure consistency and clarity.

**1. General Guidelines for All Sections:**

• Tone: Use clear, formal, and professional language. Avoid slang, and ensure that the writing is concise and to the point.

• **Formatting**:

1. Use headings for each subsection (e.g., 1. Project Overview).
2. Keep paragraphs short and readable. Use numbered lists when appropriate.
3. Ensure clarity and avoid redundancy.

**• Length:**

1. Aim for a 2-3 paragraphs for each section.
2. Be specific but avoid overwhelming the reader with unnecessary details.

**• Consistency:**

1. Make sure all sections are consistent in tone, formatting, and structure.
2. Refer to any previous sections when needed (e.g., use the same terminology, if applicable).

**2. Section-Specific Guidelines:**

**1. Project Overview and Purpose**

• **Purpose**: Introduce the project by describing its high-level objectives, goals, and expected outcomes. Explain why the project is being created and what it aims to achieve.

• **Tips**:

1. Combine the high-level project introduction with the core objectives.
2. Focus on the **why** and **what** the project is meant to accomplish.
3. Keep the content concise but clear, ensuring all major points are covered in one unified section.

**2. Requirements**

**Purpose**: Detail the functional and non-functional requirements of the project, based on the CRS (Customer Requirements Specification) document.

**Tips**:

* 1. Focus on what the system needs to do (functional) and the constraints or qualities it should have (non-functional).
  2. Be clear and specific in describing each requirement.

**3. Scope (In-Scope / Out-of-Scope)**

• **Purpose**: Clearly define the boundaries of the project by listing the tasks, features, and functionalities that are **in-scope** (included) and **out-of-scope** (excluded) from the project. This helps prevent scope creep and ensures that the project remains focused on the agreed-upon deliverables.

• **Tips**:

1. In-Scope: List all the features, tasks, or components that will be included in the project. Be as specific as possible, mentioning key functionality and major milestones.
2. Out-of-Scope: Explicitly list what is not included in the project. This may include features, tasks, or functionalities that are outside the project’s objectives, resources, or timeline.
3. Be clear and specific to avoid misunderstandings later in the project.

**4. SDLC Methodology**

• **Purpose**: Describe the Waterfall Software Development Life Cycle (SDLC) methodology being used for the project and explain why it was selected. This will guide the overall project structure, including phases, tasks, and deliverables.

• **Tips**:

1. Clearly state that the **Waterfall** methodology will be followed for this project.
2. Explain that **Waterfall** is a **sequential** model where each phase must be completed before moving on to the next, making it suitable for projects with clearly defined requirements and a structured approach.
3. Mention that this methodology ensures **predictability** and **clear milestones**, allowing for easier tracking and management of progress.

**5. Naming Conventions**

• **Purpose**: Establish a consistent naming strategy for all project files and assets to maintain clarity, traceability, and organization across the team.

• **Tips**:

1. The adopted format for naming is: filename\_FEATXXX\_XXX.
2. filename: the relevant document or file name.
3. FEATXXX: refers to the related feature number.
4. XXX: can be used for initials, versioning, or additional context.
5. Ensure all team members follow this naming convention across all documents, folders, and deliverables to maintain consistency throughout the project lifecycle.

**6. Communication Plan**

• **Purpose**: Define how the project team will communicate throughout the project to ensure effective collaboration, progress tracking, and issue resolution.

• **Tips**:

1. Specify tools and channels used (e.g., WhatsApp, Trello, GitHub, Google Docs).
2. Mention how frequently updates or meetings will happen.
3. Clarify responsibilities (e.g., who posts updates, who checks tasks, who leads meetings, etc.).

**7. Milestones**

• **Purpose**: Identify the major phases or deliverables in the project that mark significant progress points.

• **Tips**:

1. Each milestone should represent the completion of a major activity or phase (e.g., CRS Finalization, SRS Creation, Implementation Start).
2. Align milestones with your **6-week schedule**, assigning one milestone per week.
3. Make sure milestones are **measurable** and **trackable**, so progress can be clearly evaluated.

**8. Timeline**

• **Purpose**: Provide a high-level view of the project schedule, showing the duration and planned completion dates for each milestone.

• **Tips**:

1. Use a **week-based format** (e.g., Week 1: CRS, Week 2: SRS, etc.).
2. Ensure each milestone aligns with a specific week.
3. Optionally, include start and end dates for clarity.
4. Keep it simple and aligned with your Trello schedule for consistency.

**9. Work Breakdown Structure (WBS)**

• **Purpose**: Break down the entire project into manageable components or work packages. This helps clarify responsibilities and supports better planning and tracking.

• **Tips**:

1. Use **high-level divisions** based on phases (e.g., Requirement Gathering, Design, Development, Testing, Deployment).
2. Under each phase, list the major tasks involved (e.g., for Development: Frontend Coding, Backend API setup, Database Integration).
3. Since detailed tasks are already organized in Trello, this section should focus on summarizing the **main components only**.

**10. Risk Management**

• **Purpose**: Identify potential risks that might affect the project and define strategies to minimize or handle them effectively.

• **Tips**:

1. List **common project risks** (e.g., delays in deliverables, technical blockers, poor communication, data loss).
2. For each risk, briefly describe its **impact** and **mitigation strategy** (e.g., backup plans, clearer coordination, buffer time).
3. Keep it realistic and tailored to your team setup and working conditions.

**11. Configuration Management**

• **Purpose**: Describe how changes to project documents, code, and other assets will be managed, tracked, and maintained to avoid confusion or loss of progress.

• **Tips**:

* 1. Specify where files and documents will be stored and shared (e.g., Google Drive, GitHub).
  2. Define how versions of documents and code are tracked (e.g., version numbers, commit messages).
  3. Mention who is responsible for updating and organizing assets (e.g., team lead or assigned member).

**12. Change Requests**

• **Purpose**: Describe how any requested changes will be managed throughout the project to ensure controlled and documented adjustments.

• **Tips**:

1. Any requested changes should be discussed with the team before approval to assess feasibility and impact.
2. Approved changes must be properly documented and updated in Trello with revised timelines if necessary.
3. The team leader is responsible for confirming and tracking any change decisions.
4. A clear change approval process should be established to avoid scope creep.

**13. Monitoring & KPIs**

• **Purpose**: Define how project progress will be tracked and measured, focusing on key performance indicators (KPIs) to ensure project goals are met on time and within scope.

**• Tips:**

1. Use Trello boards for task tracking and weekly project updates.
2. Monitor progress by evaluating completed tasks and meeting milestones.
3. KPIs to monitor might include:
4. Milestone completion on time
5. Number of completed tasks per week
6. Quality of deliverables (measured via internal reviews)
7. Team responsiveness (how quickly team members address issues or updates)