Part 1: Reading & Analysis Task

ProjectTracker Web Application

Overview: *ProjectTracker* is a web-based project management tool designed for small teams. It allows users to create projects, assign tasks, and track progress through a user-friendly interface. The application focuses on simplicity and collaboration, making it easy for non-technical users to manage projects effectively. The product aims to streamline team communication and task management by providing a centralized platform accessible from any web browser.

Key Features:

1. **Task Management:** Users can create, edit, and delete tasks with deadlines, priorities, and labels. Tasks can be organized into project-specific lists for better tracking.
2. **Team Collaboration:** Team members can comment on tasks, attach files, and receive notifications about updates. Real-time updates ensure everyone stays informed of project changes.
3. **Dashboard Overview:** Each project has a dashboard showing overall status, upcoming deadlines, and a progress bar indicating completed tasks. This provides a quick visual summary of project health.
4. **User Roles & Permissions:** Supports multiple user roles (e.g., **Administrator**, **Project Manager**, **Team Member**) with different access levels. For example, only Administrators can create new projects or add users, while Team Members can update tasks assigned to them.

Technical Specifications:

1. **Platform:** Web application compatible with modern browsers (Chrome, Firefox, Edge). It is a responsive design, accessible on desktop and mobile browsers.
2. **Backend:** Implemented in Python using the Django framework, exposing a RESTful API for client interactions. The server logic handles business rules like permissions, task state changes, and notifications.
3. **Frontend:** Built with HTML5, CSS3, and JavaScript (React library) for a dynamic single-page application experience. The interface dynamically updates content without full page reloads.
4. **Database:** PostgreSQL is used to store data for projects, tasks, comments, and user information. The database schema is designed to maintain referential integrity (e.g., if a project is deleted, its tasks are also removed).
5. **Security:** User authentication is handled via JSON Web Tokens (JWT) over HTTPS, ensuring secure communication. Passwords are stored hashed, and the system includes role-based access control to protect sensitive actions.
6. **Performance & Scalability:** The application is tested to support **up to 10,000 concurrent users**. It uses caching for frequently accessed data (like dashboard stats) and can be deployed on cloud infrastructure with auto-scaling to handle load increases.

7. **Integration:** Provides integration endpoints for third-party tools (e.g., a calendar or email service) via its API. For instance, it can send task deadline reminders to users' email and integrate with calendar applications to show due dates.

Comprehension & Analysis Questions

1. In one sentence, explain the main purpose of the *ProjectTracker* application. Who is the target user for this product?
2. List **two** key features of *ProjectTracker* and briefly describe what each one allows the user to do.
3. How many user roles are mentioned in the specification? Name them and describe one responsibility or permission for each role.
4. Which programming language and framework are used for the backend of *ProjectTracker*? What library is used on the frontend?
5. What are two security measures mentioned in the technical specifications?
6. How many concurrent users is the application designed to support? What technique is used to help achieve this capacity?
7. What are the **main sections** of this product specification sample? List the section headings used.
8. Do you think this specification provides enough information to start developing the product? What **additional section or detail** (if any) would you add to improve it? (Explain your reasoning briefly.)

1.ProjectTracker is a web tool that helps small teams manage projects, assign tasks, and track progress.

2.Task management lets users create, edit, and delete tasks with deadlines, and team collaboration allows members to comment, attach files, and get updates.

3.There are three user roles: administrators can create projects and add users, project managers handle tasks and team progress, and team members update their assigned tasks.

4.The backend uses Python with Django, and the frontend is built with React, HTML, CSS, and JavaScript.

5. Security includes JWT authentication over HTTPS and role-based access to protect actions.

6.The app supports up to 10,000 users at the same time using caching and cloud auto-scaling.

7.The main sections are overview, key features, technical details, and security.

8.The specification is good, but adding API details and database design would help developers understand how to connect and store data.

Part 2: Writing Task – Creating a Product Specification

Use the guidance below, **draft a product specification** for your chosen product. Aim for 3-5 well-structured paragraphs (or sections), including lists or tables where appropriate. Use the *ProjectTracker* sample from Part 1 as a model for style and level of detail. **Include** the following sections in your specification:

1. **Product Name & Overview:** Start with the product name and a brief description. Explain the purpose of the product and who the target users are.
2. **Key Features:** List the main features or functionalities of your product. Use bullet points and **bold** keywords to make them clear. Provide 1-2 sentences explaining each feature and how it benefits the user.
3. **Technical Specifications:** Provide technical details about how the product is built or its components. This may include:
    a. *For software:* the platform, programming language, frameworks or libraries, database or storage system, security measures, etc.
    b. *For hardware:* physical dimensions, hardware components (CPU, memory, sensors), operating system or firmware, battery life, connectivity options, etc.
4. **User Roles/Use Cases:** Describe different types of users or give a brief example scenario of how someone would use the product.
5. **Future Improvements:** Mention features you plan to add in the future or known limitations of the current design.

Formatting & Language Tips:

1. Use clear headings for each section of your spec.
2. Present information in a logical order.
3. Use bullet points or numbered lists to break out features and technical details – this makes it easier to read.
4. Incorporate appropriate technical vocabulary
5. Write in a formal, **concise** style (similar to the sample spec), but make sure each point is understandable. Check your grammar and use of terminology to ensure precision and clarity.