Architecture Design for

ProductivityCraft

a desktop-based productivity application

Submitted to:
Dr. Amit Kumar Mondal
Associate Professor
Computer Science
and Engineering Discipline
Khulna University
Khulna

Submitted by:
M.M. Emon Hossain
Student ID: 210234
&
Sandesh Sapkota
Student ID: 210242
3rd Year, 1st Term
Computer Science
and Engineering Discipline
Khulna University
Khulna

Course No: CSE 3106

Course Title: Software Development Project

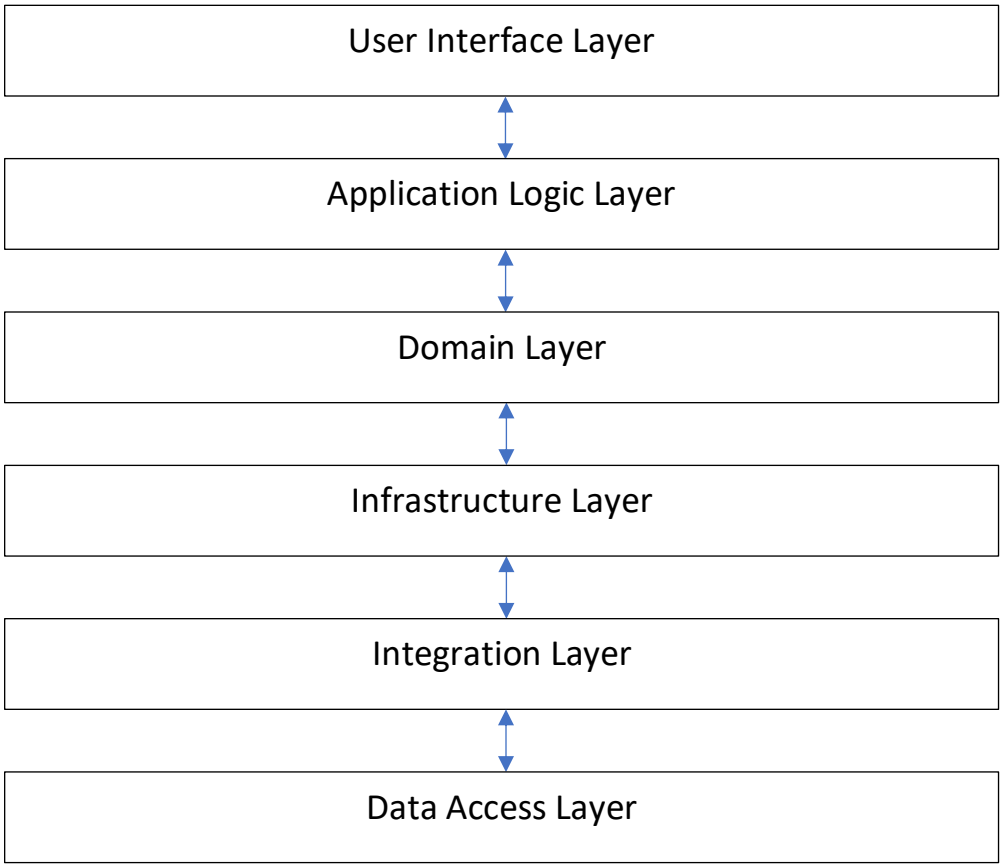
Date of Submission: February 12, 2024

Architecture Pattern: Layered Architecture

Description:

The layered architecture pattern is well-suited for ProductivityCraft due to its simplicity and effectiveness. By organizing the application into distinct layers - such as user interface, application logic, domain, infrastructure, integration, and data access - it promotes modularity, separation of concerns, scalability, flexibility, and testability. Each layer has a clear and focused responsibility, making it easier to maintain, extend, and test the application over time. This structure also allows for independent scaling of different layers based on demand, ensuring the application can adapt to changing requirements and integrate with third-party services seamlessly. Overall, the layered architecture pattern provides a solid foundation for building a robust and maintainable desktop application like ProductivityCraft.

Graphical Representation:



Explanation:

1. **User Interface Layer**: Handles how users interact with the application, including displaying information and capturing user input.
2. **Application Logic Layer**: Manages the core functionality and behavior of the application, such as processing user requests, enforcing business rules, and orchestrating the flow of data.
3. **Domain Layer**: Defines the fundamental concepts and rules of the application's domain, including entities (like tasks or users) and the logic that governs their behavior.
4. **Infrastructure Layer**: Provides the underlying infrastructure and services needed to support the application's operation, such as external services, utilities, and frameworks.
5. **Integration Layer**: Facilitates communication and interaction with external systems or services, allowing the application to exchange data and coordinate activities with other systems.
6. **Data Access Layer**: Handles the interaction with data sources such as files, including tasks like reading and writing data.