Architecture Pattern for Productivity Software

Course Title: Software Development Project
Course No. : CSE 3106

<u>Group Information:</u>

M.D. Ashiquzzaman Rahad
Student ID: 210201

Jannatul Ferdous Nijhum
Student ID: 210239

<u>Instructor:</u>
Amit Kumar Mondal
Associate Professor
Computer Science & Engineering Discipline
Khulna University,
Khulna.

Preferred Architecture

The preferred architecture for our Software project is the **Layered Architecture**

Reasoning

Layered Architecture is a popular software architecture that separates an application into distinct layers. For our Software project there are tree main layers: User interface, Application Functionality, and Locally data storing. Each layer has a specific responsibility and communicates with the others through well-defined interfaces.

The Use interface layer contains the presentation logic of our application, such as the user interface elements, the layout, and the style. The View is responsible for displaying the data to the user and receiving input. In our productivity software, the Use interface may include the windows, buttons, labels, and menus showing tasks, the timer, and the settings.

The Application Functionality is responsible for coordinating the interactions between the locally data storing and the Use interface and updating them accordingly. In our productivity software, the Controller may include the code that handles the user actions, such as adding, deleting, or completing a task, starting or stopping the timer, or changing the settings.

The locally data storing manages the data, enforces the business rules, and performs the calculations. In our productivity software, the locally data storing may include the local files and some local variables to store the required data for the tasks, the Pomodoro timer, and the user preferences. It is also concerned with the operations that effects the data.

The Layerd architecture has several advantages for our productivity software. It allows for a clear separation of concerns, which makes our code more organized, modular, and reusable. It facilitates the development and testing of each component independently, which improves the quality and reliability of our software. It enables the easy modification and extension of each component without affecting the others, which enhances the maintainability and scalability of our software. It supports the use of different technologies and frameworks for each component, which gives us more flexibility and choice in your software design.

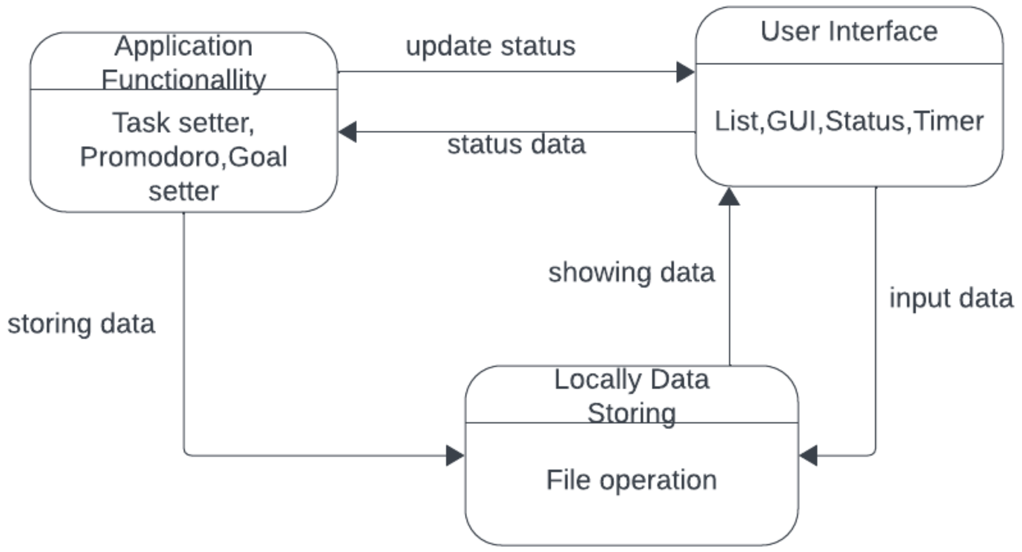

Fig: Layered Architechture for Study Outline