Hadi Abdelaal

CS-499

3/30/2024

**4-2 Milestone Three**

The Zoo Monitor System Program is the artifact that was chosen for the algorithms and data structure category. The objective of this program is to create a system for verification that controls permissions and verification for admins and zookeeper clients. As a component of the computer science seminar IT-145, the software application was conceptualized, created, and developed. The software is created in the programming language known as Java and runs on a machine with a terminal as an independent program. This artifact was chosen because it required comprehension of an application algorithm made up of two primary systems, an authentication & authorization system, and components for a system of surveillance that were enhanced. The users ought to only see information relevant to their role after they log into the application in question. The artifact is made up of architectural elements for user credential-based authentication and authorization into the monitor system as well as responsibility for the user's interactions with various module screens and actions depending on their position inside the monitor network.

The artifact entails the development of procedures for input data validation as well as default denial-aware architecture and design. This ability instills in us an attitude of security that foresees hostile vulnerabilities in software structures and designs to reveal potential weaknesses, reduce deficiencies in design, and guarantee privacy as well as enhanced information safety and resource protection. Several classes and methods are divided within the source code according to their functions and actions. We use arguments and variables in scope to demonstrate the technical implications of the connection and functioning between the various classes and methods. scanning external files into a string array variable-based flexible data structure to assess the circumstances surrounding user inputs, scanning data files line-by-line to assess conditions, and scanning information files to show their contents on the screen. The basic linear structure of data used by all of the application's classes' methods is the string array. By controlling the compromises involved in design decisions, this method enhances the development and assessment of computing solutions that use computational concepts, computer science techniques, and standards relevant to its resolution to tackle a specific issue.

Through importing from outside animal or habitat files, individuals can now list possible creatures or habitats, keep track of the daily activities of the animals under supervision, and monitor the living conditions of the animals. We demonstrate our prowess in software development, user needs analysis and interpretation, and programmatic implementation of tasks inside a structured framework. The capacity to convert a software situation into pseudocode as a single piece of program code comes from knowing the algorithms needed for it. We are able to identify the structured arrangement of a sequence of codes that may be divided into four classes and one main class. A menu display class that is repeated in each of the three main system modules makes up a single of the four classes. In order to present a header and banner, empty the shell screen, and utilize two independent classes—one for Basic colors and another for wrap lines—we incorporate GUI operations into the application's foundation. These acts illustrate our capacity to apply sound and creative technology approaches, skills, and resources to develop sustainable technology solutions that add value and achieve particular industry objectives. They also accord with the fundamentals of customer-centered design.

The source code is consistently formatted correctly, has a clear organization, and includes line breaks. According to best practices and their application in computer programming, we make use of the correct syntax and norms. The programmable data structures that are created allow for the effective use of the recorded value of variables in other class methods. Since they describe operations being carried out on a certain thing, names for methods are verbs. Dividing into methods and classes, as well as how to categorize each one and its placement when integrated into the program, results in substantial problems. We modified the software to handle problems as a result of this classification approach by figuring out if it operates through the operating system's terminal shell or via the NetBeans outputting line. In order to get our application coding to the desired point, we experimented with several code blocks to demonstrate a straightforward GUI with an attractive display. I think that’s one of the difficulties that I have to deal with while working on the enhancements for this artifact.