Thomas Hainal
CS410
2/12/2024
Hw1

1.If, hypothetically, the class sessions did not align with the Complex domain, the next best description could be Complicated, considering there might be reliance on expert knowledge (the instructor) and a pursuit of good practices in software development. To change this to a more Scrum-appropriate environment, one would encourage more self-organization within the student teams, provide clearer roles beyond Scrum Master and Product Owner, and ensure that learning objectives are achieved through iterative development with regular feedback loops, rather than through instruction alone.

2.a. The design of my specific work involved the creation of an info_nnnn.pdf file, intended to be informative and accessible to users on the professor's class website.
This PDF was formatted to ensure clarity and readability, with a simple layout with easy navigation.The PDF is linked to the class website through a few lines of HTML code, allowing for straightforward access through a clearly labeled hyperlink under the corporation which I belong to. I ensured that the file size was optimized for quick loading as the pdf was made simple with just a few lines.

b.To access the info_nnnn.pdf document, navigate to the class/course website and locate the corporations section.
Within this section, you'll find a link labeled "info_nnnn" that corresponds to the content of the PDF with the four n's being the student's last 4 id digits.
Click on the hyperlink to open the PDF in your browser where you can read, download, or print the document for your reference.
If the PDF does not load immediately, please ensure your browser is updated to the latest version or try accessing the link using a different browser.
P.S, my link is not currently working due to a permissions error.
c. Testing:
My specific link is not working on the website, but is present on the server and working, This is a defect with my html file and I am having trouble with permissions as I am getting permission denied when trying to fix it.I estimate about 15 minutes to fix this issue after asking the server admin/professor to allow permission and see what if Im lacking permissions.As it is a quick fix as my info pdf on the server is working but it is the HTML link that is not.

3 a.Levels of abstraction, these levels of hierarchy allow engineers to work on different parts of the system without needing to understand the full details of other levels, thus managing complexity and enabling more straightforward maintenance and development of the system.
User Interface Level (High-Level Abstraction):
This is the highest layer where users interact with the website through web pages. It includes the graphical layout, buttons, text, and images that the user sees and interacts with. At this

level, the complexity of underlying code and systems is hidden, providing a simple and user-friendly interface for the user to navigate the website and access linked files.

Functional Level:
Just below the user interface, this level defines what the website can do, such as serving pages, responding to user input, and handling file downloads. It abstracts the specific implementation details but provides a description of the functionality – for instance, clicking a hyperlink will request a specific PDF file from the server.

Application Logic Level:
This is where the control flow of the application is managed. It includes the server-side scripts or backend code that process user requests, handle business logic, and determine which files to serve when a link is clicked. At this level, developers work with programming languages and frameworks to implement the functionality defined above.

Data Access Level:
This level abstracts the methods and processes used to access data stores where the website's content and linked files are stored. It manages the retrieval and storage of data, such as fetching the correct PDF file when requested. It hides the specifics of the database or file storage system from the other levels.

Data Storage Level (Low-Level Abstraction):
At the bottom of the abstraction hierarchy is the actual storage of data, which could be in databases, file systems, or cloud storage. This level concerns itself with how data is physically or virtually organized, optimized for access and storage efficiency, and preserved for integrity and security.

3.b.The Stepdown Rule is a programming principle that emphasizes the importance of readability and organization in code. It suggests that code should be written so that it reads in a top-down narrative, with each function followed by those at the next level of abstraction. This means that the highest-level functions are at the top, and as you read down through the code, you “step down” one level of abstraction at a time.

The idea is to present the most important concepts first, followed by the details. This hierarchical organization helps the reader understand the high-level functionality before going into the specifics, much like reading an outline that starts with broad topics and then gets into the subtopics.
Example in the To include format:
To handle a customer order, we first verify the order details, then we process the payment, and finally, we arrange the shipment.

To verify the order details, we check the item availability and then validate the shipping address.

To process the payment, we calculate the total with taxes and then charge the customer's payment method.

To arrange the shipment, we select the appropriate courier and then print the shipping label.

3 different level one line examples of functions:
In python:
High level function:

```
def handle_customer_order(order_details):
    verify_order_details(order_details)
    process_payment(order_details)
    arrange_shipment(order_details)
```

Mid level functions:

```
def verify_order_details(details):
    check_item_availability(details)
    validate_shipping_address(details)

def process_payment(details):
    total = calculate_total_with_taxes(details)
    charge_payment_method(details, total)

def arrange_shipment(details):
    courier = select_appropriate_courier(details)
    shipping_label = print_shipping_label(details, courier)
```

Low level functions:

```
def check_item_availability(details):
    # The Code to check inventory for item availability

def validate_shipping_address(details):
    # The Code to validate the format and existence of the shipping address

def calculate_total_with_taxes(details):
    # The Code to calculate total cost including taxes

def charge_payment_method(details, total):
    # The Code to charge the customer's payment method

def select_appropriate_courier(details):
    # The Code to select the best courier based on the order details

def print_shipping_label(details, courier):
    # The Code to generate and print the shipping label
```

c.Refactoring my code above according to the writers conclusion on pp49-50, my new code is :

```
class OrderHandler:
    def process_order(self, order):
        self._review_order_details(order)
        self._execute_payment(order)
        self._initiate_shipment(order)

    def _review_order_details(self, order):
        InventoryChecker().validate_availability(order)
        AddressValidator().confirm_address(order)

    def _execute_payment(self, order):
        calculator = PaymentCalculator()
        total = calculator.compute_total_with_taxes(order)
        PaymentProcessor().bill_customer(order, total)

    def _initiate_shipment(self, order):
        courier = ShippingCoordinator().select_courier(order)
        ShippingLabelPrinter().generate_label(order, courier)


class InventoryChecker:
    def validate_availability(self, order):
        # Code to check inventory for item availability


class AddressValidator:
    def confirm_address(self, order):
        # Code to validate the format and existence of the shipping address


class PaymentCalculator:
    def compute_total_with_taxes(self, order):
        # Code to calculate total cost including taxes


class PaymentProcessor:
    def bill_customer(self, order, total):
        # Code to charge the customer's payment method


class ShippingCoordinator:
    def select_courier(self, order):
        # Code to select the best courier based on the order details
```

```
class ShippingLabelPrinter:
    def generate_label(self, order, courier):
        # Code to generate and print the
```