End of Quarter Essays

This assignment includes 5 different essays.
Each essay is worth 20 points for a total of 100 points

The purpose of each essay is for you to demonstrate your understanding of the concept and how it was applied during this course.

1. You are required to complete each of the essays prompts in this document
2. Duplication of course material, or the internet will result in the deduction of points.
3. Course material may be referenced if found in a presentation delivered in class, or part of the class reading assignments. For all referenced material provide an inline citation to the reference.
4. The essay must be in your own words, and not plagiarize any content from other students or from the internet, or any other source.
5. The essay should be in edited professional English.
6. Each essay must be a minimum 200 words, and no more than 2 pages.
7. You will use the margins, font selection, and size as established in this document. Modifications to the layout will result in reduction of points.
8. You will submit the final document as a single PDF to canvas
9. Submission must be before the posted due date, late submissions will receive a 20% penalty

1. Fundamentals of Software Engineering (20 pts)

   This class is about the software development lifecycle used by Software Engineers. During the course, we covered a product cycle from conception through deployment. Stages included identification of the product, establishment of the product vision, identification of the features, implementation, and demonstration to the stakeholders.
   During the course, three fundamental assertions were made about why Agile Methodologies is a successful Methodology in today's Software Engineering landscape. That includes Small team sizes, regular iteration with demonstratable output, and embracing change. For each of these describe the concept, provide an example of the concept in action within your team project, and what did you personally learn from the application of the concept.
   A. Small Team Sizes: Concept, Example on the project, personal learning
   B. Regular Iteration with demonstratable output: Concept, Example on the project, personal learning.
   C. Embracing Change: Concept, Example on the project, personal learning.

ANSWER:
Agile methodology is one of the simplest and most effective processes for turning a vision for a business need into software solutions. Agile is a term used to describe software development approaches that use continuous planning, improvement, team collaboration, learning, early delivery, and evolutionary development throughout a project's software development lifecycle. The Agile methodology allows for parallel development and testing, in contrast to the Waterfall process. Agile techniques aim to create the right product using tiny, cross-functional, self-organizing teams that regularly generate small functional units, allowing for frequent customer feedback and course correction as necessary. In doing so, Agile seeks to overcome the problems that traditional "waterfall" approaches of delivering massive goods over protracted periods of time experience, such as client requirements that change frequently and lead to the delivery of wrong solutions.

A. Small Team Sizes:

Concept: Agile methodology benefits small teams by allowing them to manage their work in a more efficient and effective manner. The Scrum Guide recommends a team size of three to nine people where team members can always be working on the most important tasks when using an agile approach, and they can respond quickly to changes in requirements. As stated by Jeff Bezos is well-known for his two-pizza team rule, which states that teams should be large enough to consume two pizzas. agile is well-suited to rapidly developing and releasing software products. Furthermore, the team is adaptable to changing circumstances, which is critical when working with limited resources. So, if you're part of a small team, consider using agile to your advantage.

Example of the project: We created a product called Helping hands as a group of four people, and it helps us work more productively by focusing on the most crucial project for the current sprint. We had intended to add a new feature to our product at the sprint's conclusion, but we could not achieve 100%-unit test coverage in time for another file. so, we immediately added that new feature to a backlog and got to work on the needs.

Personal learning: Agile methodology is beneficial for teams because it allows them to manage their work in a more efficient and effective way. By using an agile approach, team members can always be working on the most important tasks, and they can rapidly respond to changes in requirements. This

makes agile well-suited for quickly developing and releasing software products. In addition, the team can easily adapt to changing circumstances, which is essential when working with limited resources. Small teams could move faster, make decisions quicker and, in essence, be more agile than their larger counterparts.

B. Regular Iteration with demonstratable output:

Concept: An iterative process makes progress through successive refinement. The development team develops the first cut of the system, knowing in advance that some/many parts are incomplete. The team then iteratively enhances those parts until the product is satisfactory. With each iteration, customer feedback is considered, and the software is improved through the addition of greater detail.

Example on the project: In our project, we have eight iterations that last for one iteration i.e., one sprint for a week where we started from the basic skeleton of the website to the finished product starting from CRUDi operation, adding unique features, in each sprint and publishing at the end using Azure. Professors will give comments during the code review where improvements will be made for the next sprint and iteration continues.

Personal learning: I have learned where you break a project into timeframes called agile iterations. In agile, iterations are used to complete projects from start to finish. You will need to accomplish a distinct set of objectives and tasks during each iteration. if a project's specified goal is not attained by the end of an iteration. The following iteration will then include it. This aids the project team in making sure all the objectives are accomplished. This helps the project team ensure that all the goals are met. The aim of an agile project is to ensure that all project objectives are met and that no deadlines are missed. Agile projects depend heavily on iterations because they enable timely and high-quality project delivery.

C. Embracing Change:

Concept: One of the principles of Agile, a methodology for iterative and flexible software development, is Embrace Change. The principal says, “Welcome changing requirements, even late in development. Agile processes harness change for the customer’s competitive advantage.”

Example on the project: By revising the user stories and prioritizing the demands, we in our project accepted the adjustments in each sprint's requirements. When we were planning, our group considered adding a donation button so that users could donate directly to the specific event. However, due to schedule constraints, we changed the donation button to link to a page where users could see what could donate and how they could donate.

personal learning: I have learned that we must accept change and recognize its promise. According to the Agile concept, adjustments to technical specifications should not only be welcomed but also taken advantage of as a potential source of competitive advantage. The change enables us to discover novel alternatives and new points of view. Because we believe in the potential of this network of social entrepreneurs and meaningful firms, we at Software for Good are embracing uncertainty and mixing up our daily routines. Join us in embracing change and exploring what comes next.

2. Methodologies (20pts)

Agile is a large category applied to many practices. Consider the different practices applied during the class for the project. For each practice explain the practice, and application to the project.

From Agile (XP, Agile, Scrum, Kanban)

A. Sprints: Explanation, Application during the course.
B. Product and Sprint Backlogs: Explanation, Application during the course.
C. Kanban Board: Explanation, Application during the course.
D. Acceptance Criteria: Explanation, Application during the course.
E. Retrospectives: Explanation, Application during the course.

A. Sprints: Explanation: A project sprint in Scrum is a short period of time wherein a development team works to complete specific tasks, milestones, or deliverables. Sprints separate a project timeline into smaller, more manageable blocks By breaking your project plan into sprints, you'll set smaller goals and individual KPIs throughout, rather than waiting for the end of a project. This makes sprint project management an essential tool and skill set. A sprint cycle is a repeatable process you'll go through every time you manage and plan a sprint. There are four stages, also called Scrum ceremonies, that the project manager takes ownership of within each sprint, including sprint planning, check-ins, reviews, and retrospectives:

Application during the course: During our course time, we had eight spirits in a total of three months. We used to micro-minimize a large-scale activity into a sequence of small-scale activities without any compromise towards the product characteristics. It allows our team to participate in regular knowledge-building sessions, have efficient discussions, and focus on micro details. It can be further divided into sub-sprints, providing more focus on details pertaining to the deliverable. Also, the consistent delivery system helps in building trust with the client (professors) as they are constantly involved in the sprint review. For each sprint where our team will create a sprint goal and plan accordingly to reach the goal by making atomic cards.

B. Product and Sprint Backlogs: Explanation: The product backlog is a list that compiles all the tasks and user stories that must be done to complete the whole project. An effective product backlog breaks down each of the backlog items into a series of steps that help the development team. A product backlog is a prioritized list of steps that you need to complete to finish a project. The sprint backlog is a subset of the product backlog. The sprint backlog comes from the product backlog, but it contains only the product backlog items that can be completed during each agile sprint.

Application during the course: During the course, we used Microsoft Task planner to add a product backlog and user stories that we need for project completion. During each sprint, we will drag one task from the product backlog and creates a plan to deliver the sprint backlog to meet sprint goals.

C. Kanban Board: Explanation: A Kanban board is an Agile-based project management tool that visualizes a project's progress. Often, Kanban boards have cards that get moved across columns as tasks are completed. Kanban boards break down a project into milestones and tasks, fostering accountability and transparency in team projects. Kanban board software amplifies the power of Kanban teams by visualizing and prioritizing tasks in one central user interface. Team members know their assigned tasks and can track their progress and, on a macro level, that of their team. Because of their usefulness, Kanban boards are a common feature in project management software tools. Let's look at the top Kanban software.

Application during the course: In our project, we have created five columns in the "Product backlog, current sprint, active item, review, and done". We will move one card from the product backlog into the current sprint. Once the task is in progress our team will move a card from the current sprint into the active item, once the task is done it will move into the review. our professors will review the task and they will move to done.

D. Acceptance Criteria: Explanation: In Agile, acceptance criteria refer to a set of predefined requirements that must be met to mark a user story complete. Acceptance criteria are also sometimes called the "definition of done" because they determine the scope and requirements that must be executed by developers to consider the user story finished.

Application during the course: During the course, we created our user stories in the task board for each sprint. For each user story, we had acceptance criteria to define the scope of the user story. Example: CRUDi pages

E. Retrospectives: Explanation: An agile retrospective is an opportunity for agile development teams to reflect on past work together and identify ways to improve. Agile teams hold retrospective meetings after a time-boxed period of work is complete. During the retrospective, the team discusses what went well, what did not go as planned, and how to improve the next work period.In retrospect there are four criteria to consider they are as follows:

1. Drop
2. Add
3. Keep
4. Improve

Application during the course: During the course We created a retrospective report in which we shared the perspectives of every member regarding what we all experienced and what could be done to improve. We list "add" for additional items to add for growth, "drop" for errors to eliminate, "keep" for actions to continue taking, and "improve" for things that need to get better moving forward.

3. User Stories (20pts)
   For this quarter we focused on using exclusively user stories for all items on the tacking board. For this essay ensure you cover the following topics:

   A. Define and explain what a user story is.
   B. Provide details about the traits of a user story (use the acronym) and why each trait is important.
   C. Provide an example of a well written user story that your personally created from the tracking board for the project.
   D. For that user story explain what made it a good user story, and what changes would make it a better user story

   In Agile software development and product management User Story refers to a short, informal, and simple description of software features that are required by the end-users in the software system. Its main purpose is to provide software features that will add value to the customer's requirements. User stories are considered an important tool in Incremental software development. Mainly a user story defines the type of user, their need, and why they need that. So, in simple, a user story is a simple description of requirements that needs to be implemented in the software system. It follows a pattern:

- As a <type of user> or <person who expresses a need>
- I want/need <a goal>
- So that <a reason> or <value obtained>
  In order to determine the quality of a user narrative, it is important to remember a set of widely accepted criteria, or checklists, known by the abbreviation INVEST. An effective user story should be:

INVEST is a series of basic criteria to build quality user stories. Each criterion is necessary to build the foundation for a quality product.

- Independent: or with minimum dependency on other user stories. This facilitates the ability to execute user stories in any order since dependency naturally determines the order and also tends to become a risk.
- Negotiable: the details of user story construction emerge from a conversation (one of the 3 C's) between the person who expresses the need and the person that carries it out.
- Valuable: or with value. In other words, the result of the user story adds value to its users. In this sense, we must be aware that the concept of value can be very broad.
- Estimable: there is sufficient information to be able to guess the effort necessary to execute the user story.
- Small: the user story should not require more than a certain amount of time to be built. In teams that apply Scrum, it is common to use the convention that a story must be able to be completed during the course of a Sprint. If not, it needs to be divided. Fragmenting large user stories into smaller components is a delicate action for which other techniques can help.
- Testable: the user story contains information to make it possible to validate that the story fulfills its mission through tests or evaluations. This often requires incorporating additional information in the form of acceptance criteria, which define and clarify the scope of what you expect the story to accomplish.

The INVEST criteria are a great tool for determining the quality of the requirements.

Example of a User Story from the task board of our project: As a admin, I want to enable donate button so that users can donate".

Estimate:3

For the above-mentioned story, I have added it in an INVEST format like as an <user> I want <goal> so that <benefit>. This made it a good user story.

The above user story example follows the typical pattern and INVEST acronym that the user story follows. Here, *an admin* is the type of user in the application, to enable the donation button in the action the admin is performing on the application, and the user can get a chance to donate or desired value the user expects out of the action performed. To make this user story better, I would like to add more details of the donation that are to be added in the action part of the user story so that the user knows the outcome in detail.

4. Personal Reflection (20pts)
   Agile methods include continual improvement as a core concept. Team as well as personal retrospectives are an aspect of that. For this essay, write a personal retrospective about your contribution to the team project. In the essay, cover the format of Drop/Add/Keep/Improve, and write a detailed example for each category including the item, and example of the item and how it impacted the team, what the change would be, and an example of how the team would have operated with the expected change in place.

   A. Drop: Item, Explanation, Example, Change, Expected
   B. Add: Item, Explanation, Example, Change, Expected
   C. Keep: Item, Explanation, Example, Change, Expected
   D. Improve: Item, Explanation, Example, Change, Expected

      A. Drop: Item: Assigning a big task to one single person in a team.

         Explanation: Assigning a big task to a person instead of breaking a task into many atomic tasks made a person handle so much related to the same task.

         Example: we have added our partners to the main page as a unique feature in the sprint. That made a person concentrate on one thing instead of participating in multiple tasks.

         Change: We learned that instead of assigning a bigger task to one person, the task must be divided into numerous microtasks.
         Expected: We need

      B. Add: Item: Add comments to newly implemented functionality right after the implementation.

      Explanation: Try to add comments to newly implemented functionality right after the completion of the feature. So, that other team members can understand the code that I have added to implement that feature.

      Example When I added code for any feature implementation, I did not add the comments for that code. And it was difficult for my other team members to understand the code that I have added.

      Change: Instead of adding the comments at the end of the sprint, it is better to comment on the file right after the feature implementation

      Expected If I commented on the files right after feature implementation it would be easy for me and my team members to review and understand that code later.

      C. Keep: Item: Reviewing everyone's work before submitting

      Explanation: During the Project, in every sprint, all other team members would review everyone's work in order to avoid conflict between teams, can avoid errors, and it will be helpful to others so they can know what others have done.

Example: During one of the sprints, two members of a team worked on the same file and at the last moment it became conflicting and time waste. Both members were working on changing the background of a file which leads to conflict between members.

Change: I think before every sprint submission we need to review everyone's work so that we can avoid conflicts.

Expected: If we tried to follow the review of individual work during every sprint it would be easy to avoid errors and conflicts and less time-consuming for the team to work better.

D. Improve: Item: Improving the code at a steady pace

   Explanation: Trying to improve the coding methods and coding style will help us to improve the quality of products so that the user and other team members can understand the flow of the code and what is happening in the code.

   Example: When we are adding some pages in the CSHML file in many places we have used the "style" tag instead of creating a separate class.

   Change: Instead of adding a style tag we need to create a separate class for a better coding style and to follow coding guidelines.

   Expected: If I would have followed the code guidelines it would be easy for me to use class instead of creating a separate style tag in every file, classes can be used in many files at the same time.

5. Personal Exam Question (20 pts)
   Using the book material, create an original exam short essay question that would be appropriate for this course. The exam question should cover material from the assigned book readings, and the domain of Agile Software Fundamentals. For the exam question, provide the book reference by chapter and page, the learning objective it is pertaining to, a justification for why this is a good exam question, and an example of a valid response the question.

   A. Question:
   B. Book Reference:
   C. Learning Objective:
   D. Justification:
   E. Example Answer:

      A. Question:
         What is the difference between User Stories and Use case? What Knowledge is required for the future design?
      B. Book Reference:
         Introduction to modern software Engineering, Chapter 3, Pages no:114-120
      C. Learning Objective:
         This question will help us to understand the User Stories advantages and what knowledge is needed in order to design a future product.
      D. Justification:
         This will help us to understand how important User Stories are and what knowledge needs to be developed in order to do a better design
      E. Example Answer:
         Difference between Use case and User stories:

         - Both capture functional requirements.
         - Both are Goal Oriented and discover during user/customer workshops.
         - Easily combined with User experience activities and agile contexts.
         - User Story purpose is Customer-Orien performed perform by the user language whereas Use case is the model interaction between the user and the behavior to meet user needs and write down conditions).
         - User Stories iteration is small and the main success scenario whereas compose more stories.
         - User stories plan and estimates are via story point and velocity whereas use case will estimate project size use cases points.
         - User stories are faster and shorter verbal discussions to clarify the details whereas use case takes more time for analysis and writing, more docs model and textual model and graphical diagrams.
         - User stories acceptance tests are written on the back of the story card where use cases are created in separate docs.

         What Knowledge is required for future design?

         - USER KNOWLEDGE: Users can use user scenarios and user stories to inform the team of what users want and how they might use the software features.

- PRODUCT KNOWLEDGE: We may have experience with existing products or decide to research what these products do as part of your development process. Sometimes features must replicate existing features in these products because they provide fundamental functionality that is always required.
- DOMAIN KNOWLEDGE: This is knowledge of the domain or work area (eg., finance, event booking) that product aims to support. By understanding the domain, can think of new innovative ways of helping users do what they want to do.
- TECHNOLOGY KNOWLEDGE: New products often emerge to take advantage of technological developments since their competitors were launched. To understand the latest technology, can design features technology knowledge make use of it.