1. (**6 points John**) Describe what the term velocity means in the context of an agile software development process. Include in your description an example of velocity (**hint**: a number). Include in your description an example of using the velocity (**hint**: a formula).

- Velocity is a percentage represented by a number between 0 and 1 that describes the efficiency of your team. If velocity is 1, then your team is perfectly efficient and no time is lost. The amount of time you estimate that it will take to complete a user-story and iteration is exactly how long it will take. A default standard velocity would be 0.7. That translates to for every 10 working days, only 7 days worth of work will be completed. Velocity takes into account personal days, glitches, and other distractions that would slow down your team. Say that you originally estimate 8 days for a user-story. You estimate that your team only works about half the time they are supposed to, so your velocity is .5. To find the amount of working days it will take to complete this user-story, you need to divide your day estimate by your velocity. This comes down to $\frac{8}{0.5} = 16$. This user-story should take 8 days, but since we take into account the inefficiency of your team, we can more accurately estimate 16 working days for this user-story.

2. (**9 points John)** Describe the difference between a functional requirement and a non-functional requirement. Include in your description an example of each.

- A functional requirement is something that the program must do and the program will not work as intended if this requirement is not included, it is mandatory and verifiable. An example of a functional requirement would be that a user must complete a reCaptcha before proceeding on the website so that we can filter out malicious users. A nonfunctional requirement are specifications and restrictions on the program. These don't really affect whether or not the program will be functional, but hold it to standards and constraints other than just being executable. They have more emphasis on the quality of the program. An example of a nonfunctional requirement would be something like booting of the program should happen in 5 seconds or less. Even if it takes 6 seconds to boot, it still works just fine, but it could be better.

Use the following use case example titled "ATM Withdraw Use Case" to answer questions 3 and 4. Assume that this use case is one of many use cases and is part of a larger software engineering project. The purpose of the software engineering project is to create software for a bank automatic teller machine (ATM).

ATM Withdraw Use Case

Objective: Customer withdraws money from ATM
Flow of Events:

1. Customer selects checking option.
2. Customer enters amount to withdraw.
3. Customer confirms choice.
4. System validates the amount. If amount is not a multiple of $20, go to step 4a.
   (a) System displays error message to customer.
   (b) System returns to step 1.
5. System debits the customers account.
6. System dispenses cash.

3. (**9 points Kim**) Create three distinct functional software requirements for the ATM Withdraw Use Case.

Requirement 1: Display error message: If the customer enters an invalid number, then show the error message. (Priority 5 - 2 days)

Requirement 2: Withdraw money: if the customer enters a valid value, the customer can withdraw that amount (Priority 1 - 2 days)

Requirement 3: Show checking options: Show the options available to the customer. (Priority 1 - 2 days)

4. (**9 points Scott**) Create a use case diagram for the ATM Withdraw Use Case.

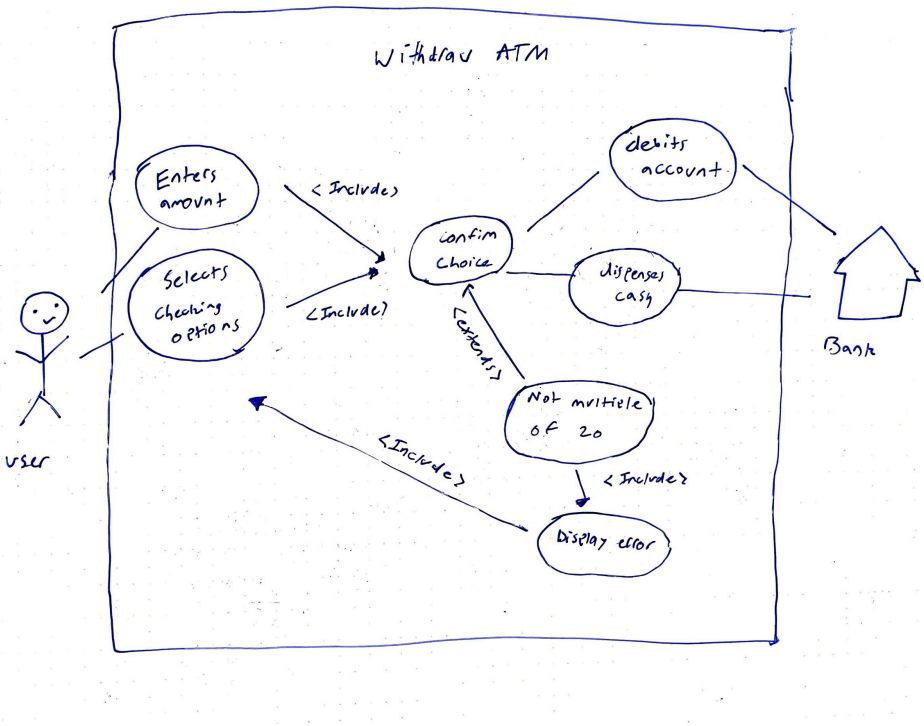
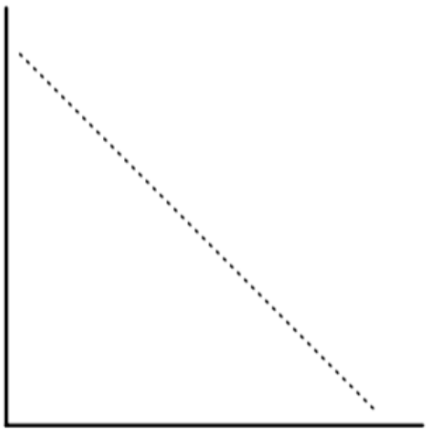


Figure 1: Burn-Down Chart



5. (Max) For the following parts of this question, refer to and modify the unlabeled Burn-Down Chart represented in Figure 1.

(a) (**3 points**) Label the X-axis and the Y-axis of the Burn-Down Chart. At a minimum, include the units of measure and where zero should be placed on each axis.

The X-axis is the days left in the current iteration, and 0 is marked on the x-axis by the intersection of the dotted line and the axis itself.

The Y-axis is the amount of work left within the current iteration; 0 is marked by the intersection of the X and Y axis.

(b) (**3 points**) The dotted line shown on the Burn-Down Chart represents the ideal burn-down rate. Describe what it means for an iteration if you are plotting your burn-down below the dotted line.

If you are below the dotted line, then you are ahead of pace. Continuing at the current pace would lead you to finish the iteration faster than expected.

(c) (**3 points**) Describe how a burn-down chart is used in a software development project.

A burn-down chart is used to monitor the progress of the project and ensure that the project is completed on time.

6. (Max) The following two parts of this question are designed to test your understanding of Software Design.

(a) (**2 points**) What information is used as input for the design phase?

The user stories.

(b) (**6 points**) Describe the difference between architectural design and detailed design in the context of software engineering.

Architectural design refers to the big picture of the project, the structure in which the project is supposed to be. Detailed design is the implementation of ideas into real code. Architectural design supports the detailed design and gives guidance to developers, while detailed design makes up what the architectural design framework.