Quiz 2

Development

- Software development cycle
    - Requirements
    - Design
    - Coding
    - Unit testing
    - Integration testing
    - Formal / acceptance testing
    - Maintenance

- Mandatory verb in a good requirement - **shall**

-

Agile

- Iterative and incremental development
- Requirements and solutions evolve via cross-functional self-organizing teams
- Adaptive planning rather than predictive planning
- Flexible response to changes
- Software delivery is the measure of progress

- MoSCoW - must have, should have, could have, won’t have

- Pros
    - Adaptive methods focus on adapting quickly to changing realities

- Cons
    - Difficult in describing exactly what will happen in the future
    - Flaws
        - Building a house without blueprint
    - Insufficient training cited as most significant cause for failed agile projects
    - Teams are not focused to meet commitments
    - Problem solving in scrum meetings can take time of too many members
    - Team members get boxed into certain roles preventing cross-training
    - Lack testing automation
    - Allows technical debt to build up if only focusing on increased functionality.

- Iterations
    - Short term frames that last 1 - 4 weeks

- **Scrum - early implementation of agile**

- Core roles
    - Product owner
        - Voice of consumer
    - Development team
    - Scrum master
        - Like PM, facilitates stuff
- **Components**
    - Sprint
        - Basic unit of development over a fixed period of time (2 weeks)
            - Planning meeting
                - tasks identified
            - Daily scrum meeting
                - What did you do, plan to do, obstacles?
            - Sprint review meeting
                - Progress reviewed
            - Result - Working product - ready to ship
                - All backlogged items implemented in Sprint
    - Sprint backlog
        - Items needed to be done prioritized by risk, business value ..
        - Contains
            - Product owner's assessment of effort
            - Development's assessment of effort
    - Velocity
        - Number of units of work / interval
    - Burndown chart
        - Chart of work left to do VS time - updated daily
    - Burn-up chart
        - Chart of work completed and total amount of work VS time, updated daily.
    - **Scrum is the manner of restarting after minor infraction**
    - Key principle
        - A customer can change their minds about what they want during development
    - **Unpredicted challenges are hard to address in a planned manner**
    - **Accept that problem cannot be fully defined**

- **User stories**
    - To capture the description of a software feature from an end-user perspective
    - Short description of something that your user will do when they come to your website
        - Displaying home screen is not a functionality in User stories
        - Login into the system is
            - Because it provides **benefit**

- Format
    - As a user, I want to …, so that …

Waterfall

- Project is divided into sequential phases, with some overlap and splashback acceptable between phases
- **Emphasis on**
    - planning,
    - time schedules,
    - target dates,
    - budgets,
    - implementation of an entire system at one time
- **Tight control**
    - is maintained by **extensive** written documentation, formal reviews
    - Is approval by user and information technology management
- Benefits
    - Time spent early can reduce costs later
    - Well suited for projects where
        - requirements and scope are fixed
        - Product is firm and stable
        - Technology is clearly understood
- Drawbacks
    - Clients may not know exact requirements until they see working software
    - Changes in requirements lead to
        - redesign
        - Development
        - Retesting
        - Long development to market timeline
    - Increased costs
- Phases
    - Conception
    - Initiation
    - Analysis
    - Design
    - Construction
    - Testing
    - Production / implementation
    - Maintenance

- **Planning poker**
    - Number of sequence - **fibonacci**
    -

- **Use cases**
    - Detailed description and the steps involved with a user's interaction with the application on how it provides one specific functionality without specifying **technology, implementation or specific user entry**
        - Displaying home screen is not a functionality
        - Login into the system is
            - Because it provides **benefit**

    - **CRUD - create, read, update, delete**
    - **Starts with - the system shall / the user shall**

    - Components
        - Title & number
        - Priority
        - Status
        - Description
        - User goal
        - Desired outcome
        - Actor
        - Dependent use cases
        - Requirements
        - Pre-condition
        - Post-condition
        - Trigger
        - **Workflow**
            - **Main difference between use cases and design use cases**
            - **In DUC, MVC should be mentioned**
        - Alternative workflow

    - MVC - model view controller
        - Software architectural pattern for implementing user interfaces

Layered architecture

- Presentation layer
    - The layer of code processing input from screens
        - Form class - validation without database access
            - Send flow of control to **action class**
        - Action class - user requested action with valid data
- Business logic
    - High level functionality invoked from presentation layer
        - Dispatch class - validation with database access
        - Manager class - manage data access objects
- Data access
    - Low level database interface methods invoked from manager layer
        - DAO class
- Database - database connectivity code

UML - unified modeling language

- Provide a standard way to visualize the design of a software system