Team Items:

Iterative Development Method: SCRUM
Estimate of Sprint Backlog Capacity: Velocity
Estimation for User Stories: Planning Poker - done as a team
Daily Commitment Meeting: Daily Standup
Responsible for Reviewing Product Backlog: Product Owner - works with the team throughout development to ensure the product sticks to what the client needs

OpenUP Phases:

Defining Project Scope: Inception -1
Managing Risks: Elaboration -2
Software Deployment: Transition -3
Development & Testing: Construction -4

Object oriented design concepts:

Single Responsibility Principle: A class should have only one job or responsibility.
High Cohesion: Classes should have closely related responsibilities.
Information Expert: The class that has the most information about a task should be the one responsible for handling it.
Low Coupling: Classes should have minimal dependencies on each other.
Law of Demeter: A class should only interact with its direct dependencies, avoiding indirect ones.
Dependency Inversion Principle: High-level modules should not depend on low-level modules, but both should depend on abstractions.
Controller: The controller is responsible for handling user input, updating the model, and selecting the view to display.
Open/Closed Principle: A class should be open for extension but closed for modification.
Polymorphism: This allows different classes to be treated as instances of the same class through inheritance.

Trello Stuff:

System Behavior Specifications: Acceptance Criteria
Feature Testing Completion Criteria: Definition of Done
Writing Functional Requirements in Agile: User Stories
Project Requirements Management Tool: Trello
Large User Story: Epic
Epic Covered in Multiple Sprints: Sprints
Task for Gathering Information: Spike
Refining User Stories for Estimation: Backlog Refinement
Initial Design Ideas for a User Story: Solution Tasks

Class Diagrams:

Class Diagram Relationship Direction: Associations
Diagram Representing Class Details: Class Diagram
Designing System Structure: Architectural Design
Understanding the Problem Space: Domain Analysis
Expert and User Daily Work Environment: Domain Model

Other:

API for Request-Response Interaction: REST
Framework for Single-Page Applications (HTML & TypeScript): Angular
Angular: Binding HTML Element/Directive Values: Property Binding
Automatically Syncing Page with App State: Data Binding
App Content Change Based on Navigation: Routing
Data and Behavior Entity: Object
Message Passing in App Components: Observables
Single-Instance Application Objects: Angular Services

First Release Required Stories: MVP
Component Testing Dependency Relationship: Seam
Feature Development Version Control: Feature Branch
Git Operation for Fetch and Merge: Git Pull
Web Structure Description Language: HTML
Common REST HTTP Operations: CRUD

User Story Template:
"As a [ROLE], I want [GOAL] so that [BENEFIT]."
Acceptance Criteria Template:
"Given [PRECONDITION], when I [ACTION], then I expect [RESULT]."