**USER Stories**

=====================================================

**USER Stories:**

1. **Identify User Roles**: Understand the different types of users who will interact with the system. This could include customers, administrators, moderators, etc.
2. **Gather Requirements**: Engage with stakeholders to gather requirements and understand the needs of each user role. This might involve interviews, surveys, or observing users in action.
3. **Write User Stories**: Use the template "As a [type of user], I want [some goal] so that [some reason]" to craft user stories that encapsulate specific features or functionalities. Ensure each story is concise, focused, and actionable.
4. **Prioritize User Stories**: Work with stakeholders to prioritize user stories based on factors such as user value, dependencies, and project constraints. This helps determine which features to tackle first.

**What is a Good User Stories?- 3C's concept**

\* Agile user stories are composed of three aspects :-

1. Card: Written description of the story, used for planning and as a reminder.

2. Conversation: Conversations about the story that serve to flesh out the details of the story.

3. Confirmation: Acceptance criteria that should be full filled and tested to ensure user require have met correctly.

**Who can create a user stories?**

1. Developer.

2. Team member.

3. product owner.->ownership will be with him

4. Some times ->scrum master.

Acceptence criteria come up with the team, product owner decide to priority or depriortise. So, acceptence criteria is with product owner.

**User Story Template:**

As a <user>, I want <this functionality to be done>, So that <we get this benefit>.

eg:-1. As a<Database Administrator>, I want <Backup data that is in database>, So that <recover information in case of data loss.>

2. As a <Backend-Developer>, I want <To build RestApi>, so that <we can communicate with frontend-app>.

3. As a <Tester>, I want<To test small modules>, so that <we can avoids faulty code/reduce bugs>.

**EPICS, AND INITIATIVES?**

* User stories are also the building blocks of larger agile frameworks like epics and initiatives.
* Epics are large work items broken down into a set of stories, and multiple epics comprise an initiative.
* These larger structures ensure that the day-to-day work of the development team (on stores) contributes to the organizational goals built into epics and initiatives.

**Example:**

* As a restaurant customer, I want to reserve a Online Truck Management System, so I can ensure I have a place my things from one place to another at my preferred time.

**Acceptance criteria**: The system should show available times and trucks for reservations. After reservation, the customer should receive a confirmation

**Four main types of user stories :**

1. **Functional User Stories**: These user stories describe features or functionalities that directly provide value to the end-user. They typically address what the system should do from a user's perspective. For example:
   * "As a customer, I want to be able to search for products by category so that I can quickly find what I'm looking for."
2. **Non-Functional User Stories**: These user stories address qualities or characteristics of the system that are not directly related to its functionality but are important for overall usability, performance, or compliance. They focus on how the system should behave or perform. For example:
   * "As a website user, I want the page load time to be under 3 seconds so that I can quickly access information without delay."
3. **User Interface (UI) User Stories**: These user stories specifically address the user interface elements and design aspects of the system. They focus on how the user interacts with the system visually and interactively. For example:
   * "As a mobile app user, I want a clear and intuitive navigation menu so that I can easily switch between different sections of the app."
4. **Exploratory User Stories**: These user stories are less defined and serve as placeholders for exploration and discovery. They are often used early in the development process to capture ideas or hypotheses that need further refinement. For example:
   * "As a content creator, I want to explore options for integrating social media sharing features into the platform to increase user engagement."

## Why are user stories important?

1. **User-Centric Focus**: User stories help keep the focus on the end-users and their needs throughout the development process. By framing requirements in terms of user goals and motivations, teams can ensure that the final product addresses real user needs and provides value.
2. **Clear Communication**: User stories provide a concise and understandable way to communicate requirements between stakeholders, product owners, and development teams. This clarity reduces the risk of misunderstandings and ensures everyone is aligned on what needs to be built.
3. **Prioritization**: User stories facilitate prioritization by allowing stakeholders to rank features and functionalities based on their importance to users. This helps teams focus on delivering the most valuable features first, maximizing the product's impact while managing constraints such as time and resources.
4. **Flexibility and Adaptability**: User stories are inherently flexible, allowing for incremental development and adaptation to changing requirements. Teams can easily add, modify, or remove user stories as they learn more about user needs or market conditions, enabling agile and responsive development.
5. **Empowerment of Development Teams**: User stories empower development teams by providing them with context and autonomy to make decisions about how best to implement features. This promotes creativity and innovation, leading to more effective solutions.
6. **Validation and Feedback**: User stories serve as a basis for validating assumptions and gathering feedback from users throughout the development process. By breaking down features into small, testable increments, teams can quickly iterate and incorporate user input, reducing the risk of building the wrong thing.