Playbook Methods Repository

# **Exploratory Testing**

Discover unexpected behaviours and interactions by testing the system outside of defined requirements. This can be done as part of the testing of individual changes, or as a separate, scheduled activity at multiple points in the development cycle.

### Remote Agility: •Medium

### Linked Tactic(s): [QA Testing Strategy](https://docs.google.com/document/u/0/d/1c5y3X6Fyh2GJo-Qs2Ssn4zFeIz9FLvA0EYONHgfMd2E/edit), [Manual Testing](https://docs.google.com/document/u/0/d/1914Lj2M2vESCNsUvRf-7OyNdhbP03C9qdNhpGEkH5rU/edit)

## Why we do it:

In Software Development, it is impossible to foresee all consequences of potential changes in advance, or have a total understanding of the system’s edge cases. Exploratory Testing aims to minimize these knowledge gaps by allowing Team Members to perform freeform testing using their knowledge of the entire system and product, or focused around error cases. In addition to uncovering gaps in business logic, Exploratory Testing can uncover issues in situations of degraded performance.

## 

## When to apply it:

* During Ticket Testing: Some amount of exploratory testing should always be considered when testing changes, especially for net-new functionality.
* As Features Near Completion: Exploratory Testing often does not provide full value until a feature is near or at completion. As an example, consider a form with multiple input fields in a web application. The tickets to add each individual field may offer only a small area for exploration; business rules around whether the field is visible, or its possible values based on the content of other fields on the form. Once the entire form has been completed and can be submitted, however, the full range of business rules can be exercised: Those internal to the form, and those from outside services receiving the form’s content. Additionally, the effects of network latency on the form and form submission can only be fully explored once the feature has been completed.

## Best Practices & Considerations:

* For scheduled sessions, have a plan for what to target, and a time box: <TBD>
* Use sessions to spread production knowledge and findings: <TBD>

## Responsible roles:

* QA Engineer: Work with Product Managers to identify the scope of each Exploratory Testing session, and carry out the testing. Document what scenarios and application areas were tested (at a level of a detail agreed upon by the team or Product Manager), and create defects for all issues encountered during the session.
* Product Manager: Work with QA Engineers to prioritize areas of investigation for each session, and when sessions should be scheduled.

## Tools:

* Online tools/platforms/services
  + xx
* Websites
  + xx
* Databases
  + xx
* Other
  + xx