# Requirement Analysis and Design

Two types of user requirements:

- Functional requirements
  - Define specific behavior or functions
  - E.g. “Ability to view farms using zoom-in, zoom-out and pan functions”

- Non-functional requirements
  - Specify criteria to judge operation of a system: E.g. Performance, Reliability, Scalability, Security, Standards
  - E.g. “Refresh time to be less than 1 second”

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

# Prototyping

- Usability: The effectiveness with which users can achieve tasks

- Usability trials: Let users use the product; gather their feedback

- Prototyping: ‘Creating a scaled-down or incomplete version of a system to demonstrate or test aspects of it’

  - Allows interaction with user
    - User is given tasks to perform using paper prototype
    - Session can be observed by people or camera

  - Helps uncover user requirements

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Paper Prototyping

- Digital prototyping
  - Draw GUI by dragging and dropping UI controls on screen
  - Writing code
- Paper prototyping
  - Much faster
  - More visual bandwidth
  - Can be done by non-technical people

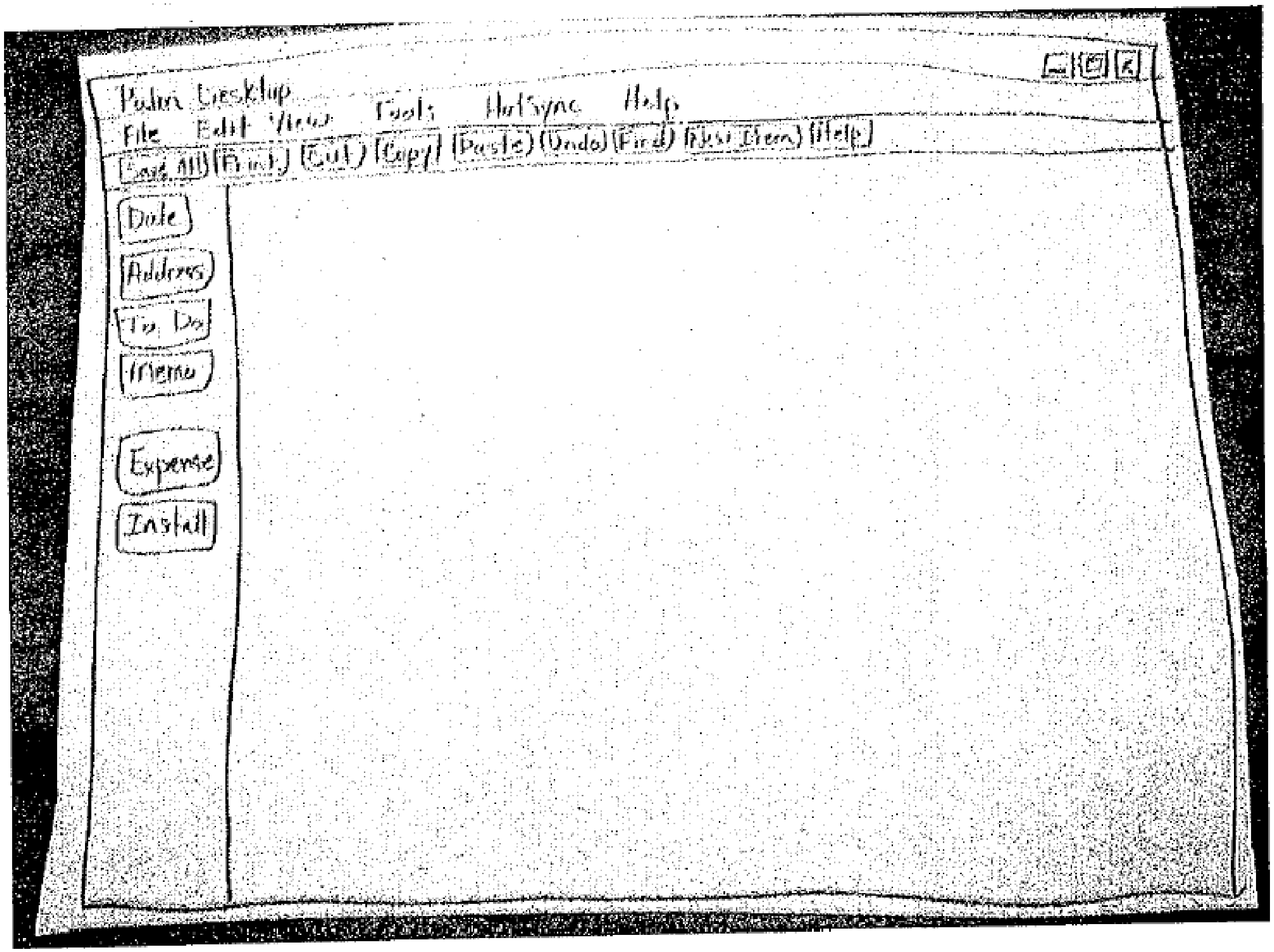
Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Creating a Paper Prototype

- Identify screens in your UI
    - Consider use cases, inputs and outputs to user
- Think about how to get from one screen to next

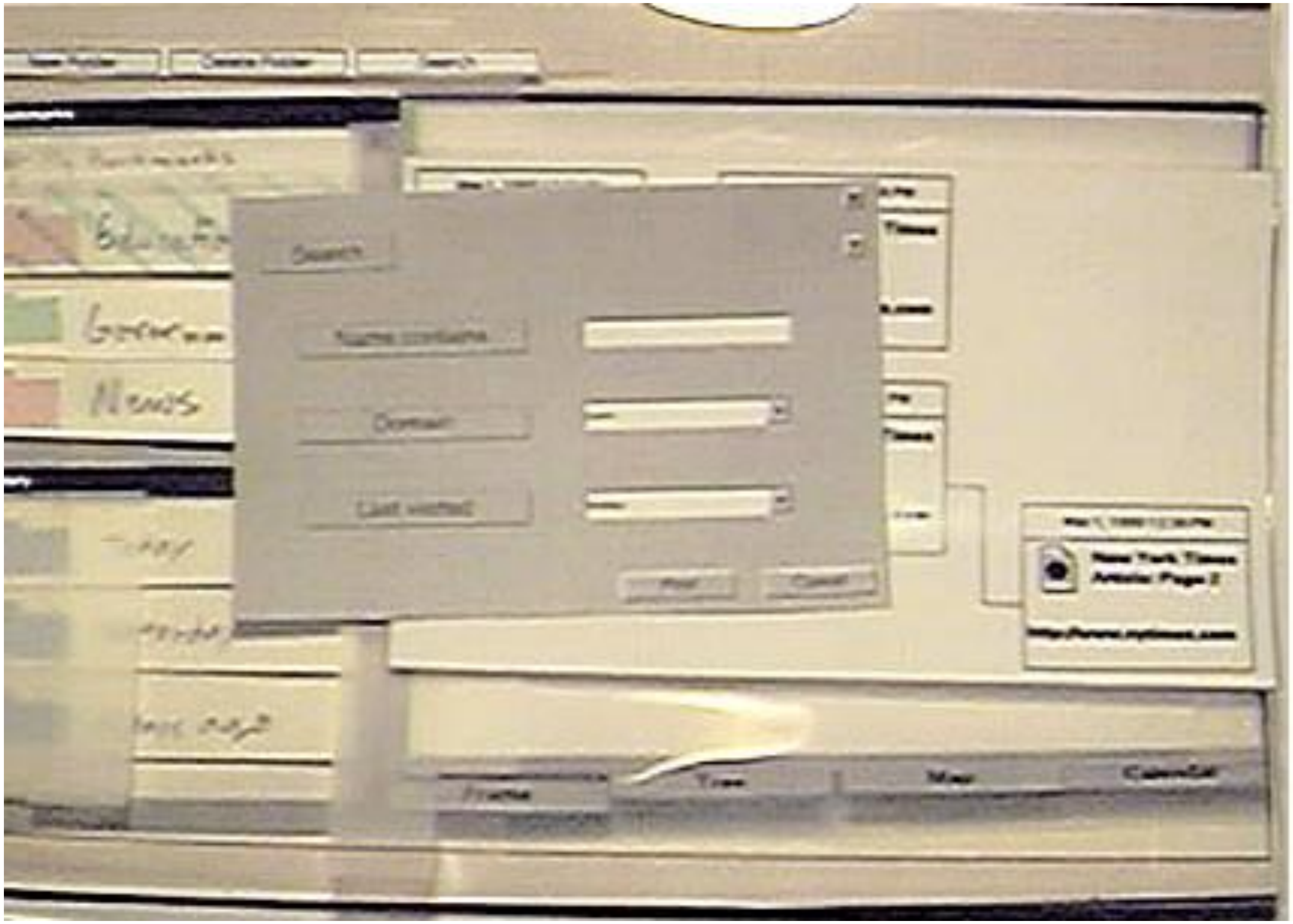
Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## E.g. Draw the App Background



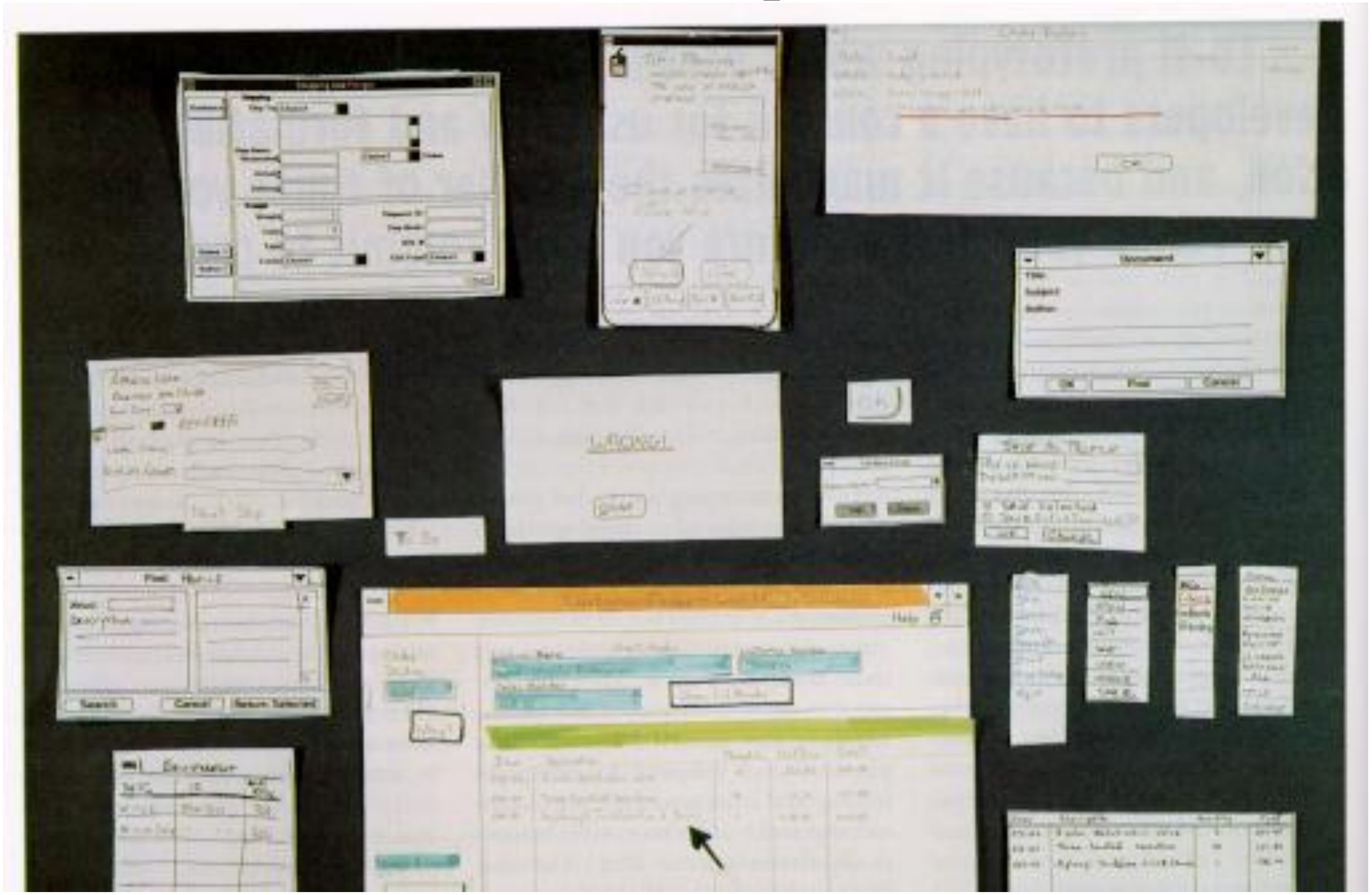
Source: University of Washington: Software Engineering course:
[http://courses.cs.washington.edu/courses/cse403/](http://courses.cs.washington.edu/courses/cse403/)
[https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/](https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/)

## …and Lay Various Sub-Screens on Top of it



Source: University of Washington: Software Engineering course:
[http://courses.cs.washington.edu/courses/cse403/](http://courses.cs.washington.edu/courses/cse403/)
[https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/](https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/)

# An Example



Source: University of Washington: Software Engineering course: http://courses.cs.washington.edu/courses/cse403/ https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Use Cases

- Help in identifying functional requirements
- Describe how a user interacts with a system to complete tasks
- Represent actor's point of view and not the developer’s
- Provide a list of things to test for

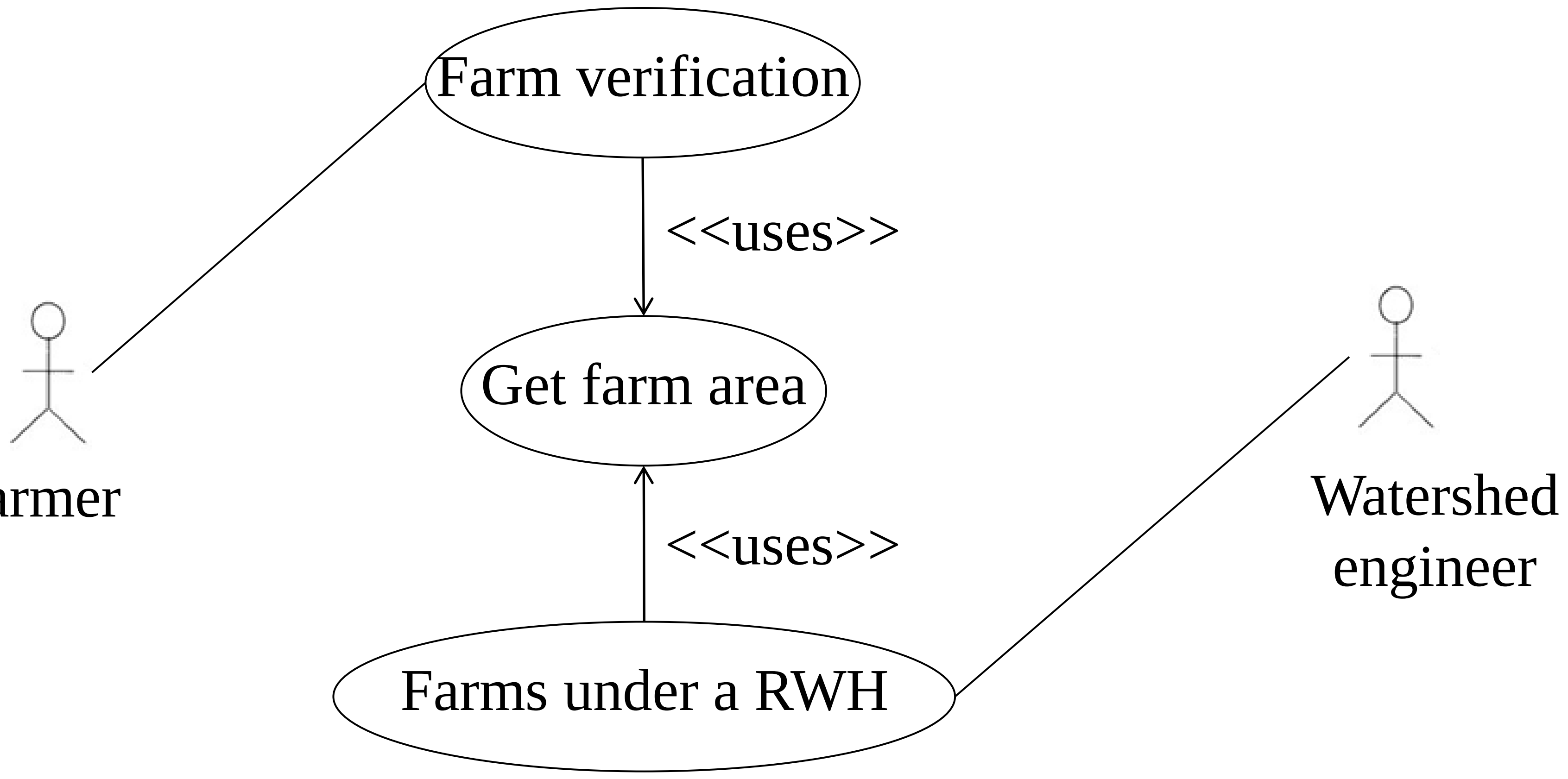
Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Users and Use Cases

| User | Use Cases |
|---|---|
| Farmer | Verify area and boundaries of a farm (own/neighbour) |
| Watershed engineer | Measure total area of all farms within certain radius of a proposed RWH system |
| Watershed manager | Measure socio-economic strata-wise area of farms within certain radius of a proposed RWH system |
| Public | Visualize change in crops of each farm over time |
| | Visualize change in depth and water yield of each well over time |

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Use Case Summary Diagram



Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Example of an Informal Use Case

Farmer verifies area and boundaries of a farm.

The farmer locates a farm on the map visually using landmarks such as roads, trees, etc. Pans and zooms to it as required. Gets its area and checks with own data. Verifies boundaries with neighbours and also their farm areas.

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Example of a Formal Use Case

"Verify area and boundaries of a farm" (Farmer)

Main (success) scenario *(using 3-6 action steps)*

1. Farmer locates farm on the map visually.
2. Gets its area and verifies it.
3. Views and verifies boundaries with neighbours. Gets and verifies their farm areas too.

(Failure) Extensions

1. Unable to locate a farm visually *(condition)*
    a. Farmer uses <u>query-by-name</u> *(action steps)*
    *(can call another use case)*

Source: University of Washington: Software Engineering course: [http://courses.cs.washington.edu/courses/cse403/](http://courses.cs.washington.edu/courses/cse403/) [https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/](https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/)

## Test Case

- Name of what is being tested
  - E.g. Query-by-name of an existing farmer
- Input arguments
  - E.g. Farmer name
- Expected output
  - E.g. Selection of a farm whose owner’s name is same as input argument provided
- Actual output produced
  - E.g. Farmer name associated with the selected farm, if any

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

# List of submissions for each group project

1. List of users, use cases and test cases

| Sl. No. | User | Use cases | Test cases |
|---|---|---|---|
| 1 | … | … | … |

2. Use case summary diagram
3. For each use case:
   i. Informal use case
   ii. Formal use case
      a. Success scenario
      b. Failure extensions
4. For each test case:
   i. Inputs by tester
   ii. Expected outputs

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/

## Test-driven development

- "Testing in Django": https://realpython.com/blog/python/testing-in-django-part-1-best-practices-and-examples/

- "Test-Driven Development with Python" by Harry Percival (O'Reilly, 2014) (https://www.obeythetestinggoat.com/book/preface.html#_why_i_wrote_a_book_about_test_driven_development) (https://www.obeythetestinggoat.com/pages/book.html#toc)

Source: University of Washington: Software Engineering course:
http://courses.cs.washington.edu/courses/cse403/
https://courses.cs.washington.edu/courses/cse403/08wi/lectures/slides/