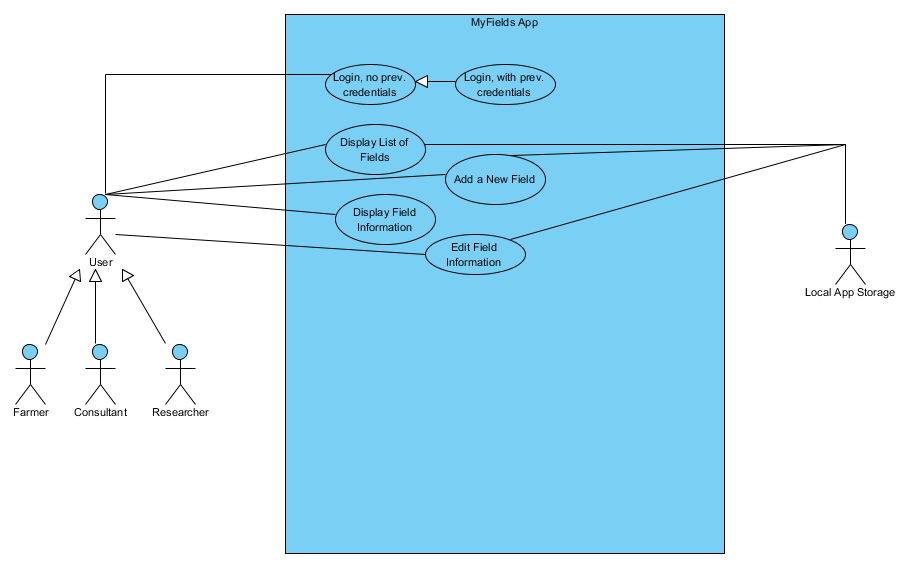
# Requirements

## Use Case Diagram



## Use Case Descriptions

### 1.2.1. Login, no previous credentials

**Actors**: User

**Stakeholders and Needs**:

User – Login to system to interact.

**Preconditions**: User has started system.

**Postconditions**: Fields List is displayed.

**Trigger**: User starts system.

**Basic Flow**:

1. System displays form with inputs for username and password, and login button
2. User inputs username and password
3. User clicks “Login” button
4. System authenticates with iWheat.org to retrieve user’s fields and associated info
5. System stores retrieved info and displays it in Fields List page

**Extensions**:

4a. If user credentials are invalid, return to step 1 and display “Invalid username/password.”

### 1.2.2. Login, with previous credentials

**Actors**: User

**Stakeholders and Needs**:

User – Login to system to interact.

**Preconditions**: User has started system, and has logged in to system previously.

**Postconditions**: Fields List is displayed.

**Trigger**: User starts system.

**Basic Flow**:

1. System retrieves credentials from secure local storage
2. System authenticates with iWheat.org to retrieve user’s fields and associated info
3. System checks retrieved info against locally stored info
4. System updates local storage to the most recent info and displays Fields List page

**Extensions**:

2a. If user credentials are invalid, go to step 1 of use case 1.2.1.

4a. If information does not match, fields have been updated while app was closed, via the website. Website information should take precedence.

### 1.2.3. Display List of Fields

**Actors**: User

**Stakeholders and Needs**:

User – View list of all fields, scroll through list

**Preconditions**: User is logged in to the system

**Postconditions**: User sees list of fields displayed

**Trigger**: User clicks “Login” with valid credentials OR valid credentials have been detected from a previous run

**Basic Flow**:

1. System uses user credentials to retrieve user’s field information from the website.
2. System parses retrieved information into Fields objects.
3. System checks local storage for Fields objects.
4. System builds and displays a form with each of those Fields objects to the user.

**Extensions**:

3a. If local fields objects are detected, and they do not match, the more recent object should be written into local storage.

### 1.2.4. Add a New Field

**Actors**: User

**Stakeholders and Needs**:

User – Add a new field to the list of user’s fields

**Preconditions**: User is logged in to the system

**Postconditions**: User has a new field displayed in their list of fields.

**Trigger**: User clicks on the “Add Field…” button on the Fields List page, or “Add a Field” from the Pest Sampler “Select a Field” page.

**Basic Flow**:

1. System builds and displays a form asking for each piece of information necessary to build a new field, with buttons for “Save” and “Cancel”
2. User inputs required information, and any optional information.
3. User clicks “Save” or “Cancel.”
   1. If User clicks Save, a new Field object is created from the entered information
   2. If User clicks Cancel, “Are you sure?” is displayed in a message box with “Yes” or “No” options.
      1. If “Yes” is pressed, information is discarded and Fields List is displayed
      2. If “No” is pressed, User returns to previous form.
4. System builds and displays the updated form with each field, including the new, to the user.

**Extensions**:

1a. ABSOLUTE necessary information is currently: Field Name, Latitude, Longitude

1b. Additional optional information: Size (acres), Soil Type, Tillage System, Irrigation Type

3a. Field Name, Latitude, Longitude, and Size should be validated. Other values are dropdowns.

### 1.2.5. Display Field Information

**Actors**: User

**Stakeholders and Needs**:

User – View information related to a specific field.

**Preconditions**: User is logged in to the system

**Postconditions**: User sees all information related to a specific field

**Trigger**: User clicks on a specific field in the Fields List OR clicks a link to that field from a page related to that field.

**Basic Flow**:

1. System retrieves the Field object associated with the named field.
2. System builds and displays a form containing all information associated with that field, including previous pest samples.

**Extensions**:

2a. Information to Display: Field Name, Latitude, Longitude, Size (acres), Soil Type, Tillage System, Irrigation Type

### 1.2.6. Edit a Field

**Actors**: User

**Stakeholders and Needs**:

User – Edit information associated with a field.

**Preconditions**: User is logged in to the system, and is on the desired field’s information page

**Postconditions**: System updates field information.

**Trigger**: User clicks on the “Edit Field” Button on the Field Information page.

**Basic Flow**:

1. System modifies the Field Information form to have editable fields, and a “Save” and “Cancel” button.
2. User clicks on the desired area to edit, and inputs the desired information.
3. User clicks “Save” or “Cancel.”
   1. If User clicks Save, that field object is updated with the new information.
   2. If User clicks Cancel, “Are you sure?” is displayed in a message box with “Yes” or “No” options.
      1. If “Yes” is pressed, information is discarded and Fields Information page is displayed.
      2. If “No” is pressed, User returns to previous form.
4. System returns to the Field Information form, which is updated if necessary.

**Extensions**:

1a. Necessary information is currently: Field Name, Latitude, Longitude, Size (acres), Soil Type, Tillage System, Irrigation Type

3a. Field Name, Latitude, Longitude, and Size should be validated. Other values are dropdowns.