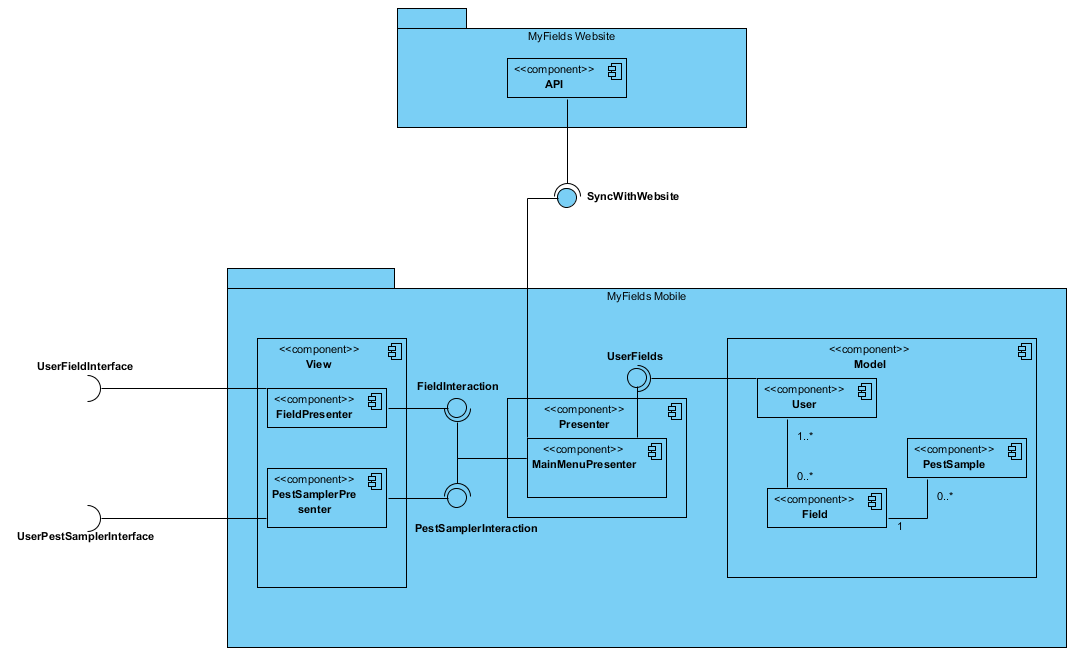
# 3. Component Diagram & Analysis

## Component Diagram



## Component Analysis

This component diagram gives a visual representation of the responsibilities of each piece of the MyFields Mobile website. Each section has been lumped into components in the Model-View-Presenter design schema to aid in visualizing the breakdown of responsibilities in this design. As noted, the FieldPresenter and PestSamplerPresenter both act as part of the view, interfacing with the user. The main presenter component is the MainMenuPresenter, which acts as both an interface between the Model and the View, and as an interface to synchronize data with the website. The Model consists of the necessary user information and potentially preferences, as well as that user’s field information and those fields’ associated pest samplings.

### The View

The View component acts as the user interface for the system. The current design has two main functions: to display, manage, edit, and add new fields; and to create pest samples associated with those fields. Thus, to make for a modular system, the View component needs two different interfaces. These interfaces are provided by the FieldPresenter and PestSamplerPresenter components. By breaking the view into components for each function of the app, it makes it easy to extend the app in the future; simply add a new user interface in the view, add any necessary components to the Model data, and link them with the Presenter.

#### Interfaces

The View component has four interfaces; 2 per inner component. The first interface is the UserFieldInterface for field information. This interface should provide a means for the user to add, edit, and view information about each of their fields individually, or a list of the fields overall. The second interface is from the FieldPresenter to the MainMenuPresenter; called FieldInteraction. This interface should provide all necessary information about one particular field, or a list of all fields in general, for the FieldPresenter to display to the user.

The third interface is the UserPestSamplerInterface for creating pest samples. This interface should provide a means for the user to create new pest samples to link to the fields. The fourth interface is the PestSamplerInteraction interface between the PestSamplerPresenter and the MainMenuPresenter. This interface should provide a field index to link to the Pest Sample, and should return a created pest sample for the Presenter to add to the Model, linked to the Field provided.

### The Presenter

The Presenter component acts as the interface between the Model and the View components. By decoupling these, this makes the overall system more modular, and requires less modification when extending the app into the future. The Presenter component will also act as an interface between the MyFields website and the Model storage, synchronizing the data stored between the two once per hour. This functionality adds substance to the Presenter class, so it is more than just a go-between for the View and Model.

#### Interfaces

The Presenter has four interfaces, and is the most highly coupled component in the system. The first interface must be from the MainMenuPresenter component to the User component in the Model. See [section 3.2.3.1](#_Interfaces) for information about this interface. The second interface must be from the MainMenuPresenter to the API of the MyFields website. This interface will synchronize the data contained within the Model to the information contained by the website about that user. The third and fourth interfaces are very similar. The third is to the FieldPresenter component of the system from MainMenuPresenter; see [section 3.2.1.1](#_Interfaces_1) for information about this interface. The fourth is from MainMenuPresenter to the PestSamplerPresenter in the View component. See [section 3.2.1.1](#_Interfaces_1) for information about this interface.

### The Model

Currently, there are only three types of data that need to be stored by the app. The system must have a user, that owns a set of fields (a user may have no fields), with a potential set of preferences for the app. The app must also store all information about that user’s fields. This should only be current information about the fields, as the website will store all previous field information. The last piece of data stored in the model is the Pest Samples. These objects are associated with a particular field, and contain information such as the cost of the field’s crop, the cost to treat for that sample’s pest, and a recommendation of whether to treat the field or not. A new pest sample will be built each time the Pest Sampler is used.

#### Interfaces

The Model only has one interface. This interface is provided by the Model component (specifically the user) to the Presenter component; specifically the MainMenuPresenter. This interface must provide the MainMenuPresenter with the Model’s list of fields, their associated Pest Samples (which should be packaged in the field) and any user preferences related to the request by the MainMenuPresenter.