RentIt's Plant Catalog & Purchase Orders

RentIt is a successful equipment rental company that operates in the Baltic countries. For the time being, however, RentIt relies on a human-operated workflow to keep track of customer relations and services. In that context, you have been hired to implement a web-based front-end for an initial automated version of their business process. Their aim is, in a first stage, to provide a web-based Plant catalog and an on line form to create plant hire requests (which we will refer with the generic name of “Purchase Orders”).

To illustrate the requirements to be implemented in the first release of the prototype, we will use a sequence of wireframes. First, the user can query the system to retrieve the list of plants that are available for a given rental period. To this end, the user enters the name of the equipment to be hired (referred to as a “plant”) and the start and end dates for the rental. The corresponding form should look like:

When the user clicks the “Query plant catalog” button, the system retrieves from the underlying database, the list of plants matching the requirements. Of course, one of the requirements is that a plant, to be part of the result, should not be engaged for rental with another customer for the same period. An example of the result of the catalog query is shown in the second wireframe.
Then the customer selects the plant that fits better her/his requirements from the list by clicking the corresponding link. For the sake of simplicity, we will assume that a purchase order (aka as plant hire request) is created for a single plant. Therefore, for the first prototype a Purchase order will be created immediately after selecting one row (link “Select”) from the list of available plants. Then, the system must present the details of the newly created purchase order, as shown in the last wireframe:

For your database schema, you can use as reference the following UML class diagram. Of course, you are free to propose another design. Note that the class "Customer" is not required for this version of the prototype.

A Gherkin user story capturing two scenarios (similar to the one illustrated with the sequence of wireframes) can be found at: https://goo.gl/9bD2gk. In addition to the user story, you will find three HAML templates that you can optionally use in your implementation.

For grading purposes, you are required to follow the BDD-TDD cycle and to record every step in the cycle in your Bitbucket repository. The task breakdown and corresponding marks are as follows:

- (BDD) Implementation of all cucumber steps 15 pts
- (TDD) Controller and model rspec tests 10 pts
  At least one controller test for the case where a plant is not available when the customer attempts to create the purchase order (e.g. the plant has been reserved by another customer) and one model test, e.g. on a method that checks whether a plant is available or not.
- Database related tasks (at least one database migration) 10 pts
- Evidence that you followed the “BDD-TDD cycle” approach 5 pts
  Commit every phase in the “BDD-TDD cycle” (one point will be deducted for each step that is found test and code updated in the same commit).