```
<script type="text/ng-template" id="myModalContent.html">
    <div class="sarah-modal-body">
        <a class="evolbclose" ng-click="mc.cancel()">X</a>
        <div class="text-center">
            <h1 style="color:red">{{mc.modalContent.name}}</h1>
            <img src="{{mc.pictureContent}}" class="modal-cover-photo" />
        </div>
        <div class="text-center">
            <h2 style="color:red">Congrats! You have discovered {{mc.modalContent.name}}!</h2>
        </div>
        <div class="form-group has-feedback myModal">
            <label class='edited pull-left' id='edited_categoryaddplaces'>Category:</label>
            <select name='Category' class='form-control' id='category'
                    data-ng-model="mc.selectedCategory"
                    ng-options="category as category.tagName for category in mc.categoryContent" ng-change="mc.onCategoryChange()" required></select>
        </div>
        <div class="form-group has-feedback myModal">
            <label class='edited col-lg-12 myModal'>Attribute:</label>
            <select class="form-control col-lg-6" ng-options="attrib.attributeName for attrib in mc.attributes track by attrib.attributeName" ng-model="mc.attribute"></select>
            <button type="button" class="btn btn-danger" ng-click="mc.addAttribute()"><i class="fa fa-plus"></i> Add</button>
        </div>
        <div class="form-group has-feedback myModal" ng-repeat="attrib in mc.addedAttributes track by attrib.attributeName">
            <label class="edited">{{attrib.attributeName}}</label>
            <input type="text" class="form-control" ng-if="attrib.attributeType === 0" ng-model="attrib.value" />
            <input type="number" class="form-control" ng-if="attrib.attributeType === 1 || attrib.attributeType === 2" ng-model="attrib.value" />
        </div>
        <div class="form-group has-feedback myModal">
            <label class='edited pull-left' id='edited_tagsaddplaces'>Tags:</label>
            <select name='Tags' class='form-control' id='tagsSelect'
                    chosen
                    multiple
                    data-ng-model="mc.selectedFreeTag"
                    ng-options="tag as tag.tagName for tag in mc.freeTagContent"></select>
        </div>
    </div>
    <div class="modal-footer">
        <button type="button" class="btn btn-danger" ng-click="mc.save()">Save</button>
    </div>
</script>
```

```
<script>
    (function () {
        "use strict";

        angular.module(APPNAME)
            .controller('modalController', ModalController);

        ModalController.$inject = ['$scope', '$baseController',
          '$uibModalInstance', 'modalContent', 'pictureContent',
          'categoryContent', 'freeTagContent', 'cityContent', '$placesService',
          '$utilityService', '$entitiesService']

        function ModalController(
            $scope
            , $baseController
            , $uibModalInstance
            , modalContent
            , pictureContent
            , categoryContent
            , freeTagContent
            , cityContent
            , $placesService
            , $utilityService
            , $entitiesService) {

            var vm = this;

            vm.$entitiesService = $entitiesService;

            $baseController.merge(vm, $baseController);

            vm.$scope = $scope;
            vm.notify = $entitiesService.getNotifier($scope);
            vm.$uibModalInstance = $uibModalInstance;

            vm.selectedCategory = null;
            vm.selectedFreeTag = null,

            vm.modalContent = modalContent;
            vm.pictureContent = pictureContent;
            vm.categoryContent = categoryContent;
            vm.freeTagContent = freeTagContent;
            vm.cityContent = cityContent;
            vm.onCategoryChange = _onCategoryChange;
            vm.addAttribute = _addAttribute;

            function _onCategoryChange() {
                vm.addedAttributes = [];
                vm.attributes = [];

                return vm.$entitiesService.getEntityAttributes
                  (vm.selectedCategory.tagName).success(onGetEntityAttributes);
            }

            function onGetEntityAttributes(data) {
                var items = data.items;

                vm.notify(function () {
                    vm.attributes = items;
                });
            }

            function _addAttribute() {
                if (vm.attribute && vm.addedAttributes.indexOf(vm.attribute) === -1) {
                    vm.addedAttributes.push(vm.attribute);
                }
            }

            vm.save = function () {

                var address_components = vm.modalContent.address_components;
                var components = {};

                $.each(address_components, function (k, v1) {
                    $.each(v1.types, function (k2, v2) {
                        components[v2] = v1.short_name
                    });
                })

                var tagIds = $.map(vm.selectedFreeTag || [], function (v) {
                    return v.id;
                });

                var placeSlug = $utilityService.toSlug(vm.modalContent.name);

                if (vm.modalContent.price_level == null) {

                    vm.modalContent.price_level = 0;

                } else { }


                console.log("Components >>", components);

                var placesObject = {

                    Name: vm.modalContent.name,
                    Description: vm.modalContent.name,
                    OperatingHours: vm.modalContent.opening_hours ? vm.modalContent.opening_hours.weekday_text.join('\n') : "" || "Everyday",
                    PhoneNumber: vm.modalContent.formatted_phone_number,
                    Website: vm.modalContent.website,
                    CategoryId: vm.selectedCategory.id,
                    TagIds: tagIds,
                    Slug: placeSlug + "-" + components.postal_code,
                    ExtPlaceId: vm.modalContent.place_id,
                    CityId: vm.cityContent || 33,
                    Price: vm.modalContent.price_level,
                    Address1: components.street_number + ' ' + components.route,
                    City: components.locality || components.sublocality,
                    State: components.administrative_area_level_1,
                    ZipCode: components.postal_code,
                    Latitude: vm.modalContent.geometry.location.lat(),
                    Longitude: vm.modalContent.geometry.location.lng(),
                    Url: vm.pictureContent
                };

                console.log("PAYLOAD OBJECT>>>", placesObject);
                $placesService.insert(placesObject, onSubmitPlace, onSubmitPlaceError);


            };

            function onSubmitPlace(data) {
                console.log("Post Success:", data)

                vm.existingId = data.item.id;
                $entitiesService.addRecord(vm.selectedCategory.tagName, data.item.id).success(onAddRecord);
                vm.$alertService.success(data.item.name + " has been discovered!", "New Place Added!");
                vm.$uibModalInstance.close();
            }

            function onAddRecord() {
                console.log("Record Added");

                var values = $.map(vm.addedAttributes, function (v) {
                    return {
                        RecordId: vm.existingId,
                        AttributeId: v.attributeId,
                        Entity: vm.selectedCategory.tagName,
                        Value: v.value
                    };
                });
                $entitiesService.addAttribtueValues(values);
            }

            function onSubmitPlaceError(data) {
                console.log("ERROR");
                vm.$alertService.error("Place may already exist or city is not selected!", "Error!");
            }

            vm.cancel = function () {
                vm.$uibModalInstance.dismiss('cancel');
            };
        }
    })();
</script>
```