```
diff --git a/Gemfile b/Gemfile
index 4030acd..0a96a61 100644
--- a/Gemfile
+++ b/Gemfile
@@ -50,7 +50,8 @@ group :test do
  gem 'capybara', '>= 2.15'
  gem 'selenium-webdriver'
  # Easy installation and use of chromedriver to run system tests with Chrome
-  gem 'chromedriver-helper'
+  #gem 'chromedriver-helper'
+  gem 'webdrivers', '~> 3.0'
 end

 # Windows does not include zoneinfo files, so bundle the tzinfo-data gem
diff --git a/Gemfile.lock b/Gemfile.lock
index e20b283..e80457c 100644
--- a/Gemfile.lock
+++ b/Gemfile.lock
@@ -44,8 +44,6 @@ GEM
      tzinfo (~> 1.1)
    addressable (2.7.0)
      public_suffix (>= 2.0.2, < 5.0)
-   archive-zip (0.12.0)
-     io-like (~> 0.3.0)
    arel (9.0.0)
    bindex (0.8.1)
    bootsnap (1.4.5)
@@ -62,9 +60,6 @@ GEM
      xpath (~> 3.2)
```

```
    childprocess (2.0.0)
      rake (< 13.0)
-   chromedriver-helper (2.1.1)
-     archive-zip (~> 0.10)
-     nokogiri (~> 1.8)
    coffee-rails (4.2.2)
      coffee-script (>= 2.2.0)
      railties (>= 4.0.0)
@@ -82,7 +77,6 @@ GEM
      activesupport (>= 4.2.0)
    i18n (1.6.0)
      concurrent-ruby (~> 1.0)
-   io-like (0.3.0)
    jbuilder (2.9.1)
      activesupport (>= 4.2.0)
    loofah (2.2.3)
@@ -175,6 +169,10 @@ GEM
      activemodel (>= 5.0)
      bindex (>= 0.4.0)
      railties (>= 5.0)
+   webdrivers (3.9.4)
+     nokogiri (~> 1.6)
+     rubyzip (~> 1.0)
+     selenium-webdriver (~> 3.0)
    websocket-driver (0.7.1)
      websocket-extensions (>= 0.1.0)
    websocket-extensions (0.1.4)
@@ -188,7 +186,6 @@ DEPENDENCIES
  bootsnap (>= 1.1.0)
```

```
  byebug
  capybara (>= 2.15)
-  chromedriver-helper
  coffee-rails (~> 4.2)
  duktape
  jbuilder (~> 2.5)
@@ -201,6 +198,7 @@ DEPENDENCIES
  tzinfo-data
  uglifier (>= 1.3.0)
  web-console (>= 3.3.0)
+  webdrivers (~> 3.0)

 RUBY VERSION
   ruby 2.5.3p105
diff --git a/app/controllers/cars_controller.rb b/app/controllers/cars_controller.rb
index b6dff67..1fa9367 100644
--- a/app/controllers/cars_controller.rb
+++ b/app/controllers/cars_controller.rb
@@ -1,6 +1,12 @@
 class CarsController < ApplicationController
  before_action :set_car, only: [:show, :edit, :update, :destroy]

+  #Implement cars search
+  def search
+    @cars = Car.where("model like :search OR vin like :search", search:"%#{params[:query]}%")
+    render :index
+  end
+
  # GET /cars
```

```
  # GET /cars.json
  def index
@@ -10,6 +16,7 @@ class CarsController < ApplicationController
  # GET /cars/1
  # GET /cars/1.json
  def show
+    @cars = Car.find(params[:id]) #show car
  end

  # GET /cars/new
@@ -27,6 +34,7 @@ class CarsController < ApplicationController
  # POST /cars.json
  def create
    @car = Car.new(car_params)
+    @parts = Part.all # added to prevent create from breaking when no values entered

    respond_to do |format|
      if @car.save
diff --git a/app/controllers/makes_controller.rb b/app/controllers/makes_controller.rb
index 19460d9..2bdba61 100644
--- a/app/controllers/makes_controller.rb
+++ b/app/controllers/makes_controller.rb
@@ -1,6 +1,12 @@
 class MakesController < ApplicationController
  before_action :set_make, only: [:show, :edit, :update, :destroy]

+  # Implement make search
+  def search
+    @makes = Make.where("name like ?", "%#{params[:query]}%")
```

```
+    render :index
+  end
+
   # GET /makes
   # GET /makes.json
   def index
@@ -10,6 +16,7 @@ class MakesController < ApplicationController
   # GET /makes/1
   # GET /makes/1.json
   def show
+    @makes = Make.find(params[:id]) #show make
   end

   # GET /makes/new
diff --git a/app/controllers/parts_controller.rb b/app/controllers/parts_controller.rb
index 2e52866..2b7e5a3 100644
--- a/app/controllers/parts_controller.rb
+++ b/app/controllers/parts_controller.rb
@@ -1,6 +1,12 @@
 class PartsController < ApplicationController
   before_action :set_part, only: [:show, :edit, :update, :destroy]

+  # Implement parts search
+  def search
+    @parts = Part.where("name like ?", "%#{params[:query]}%")
+    render :index
+  end
+
   # GET /parts
```

```
  # GET /parts.json
  def index
@@ -10,6 +16,7 @@ class PartsController < ApplicationController
  # GET /parts/1
  # GET /parts/1.json
  def show
+    @parts = Part.find(params[:id]) #show part
  end

  # GET /parts/new
diff --git a/app/models/car.rb b/app/models/car.rb
index b636ddc..b3b7242 100644
--- a/app/models/car.rb
+++ b/app/models/car.rb
@@ -1,4 +1,9 @@
 class Car < ApplicationRecord
  belongs_to :make
  has_and_belongs_to_many :parts
+  validates :make, presence: true # validate make is present
+  validates :model, presence: true # validate model is present
+  validates :vin, uniqueness: true, presence: true, format: {with: /\A[A-Z0-9]*\z/, message: "may only contain A-Z and 0-9"} # validate vin
+  validates_associated :parts # validate parts exist
+  validates :parts, presence: true # validate parts are present
 end
diff --git a/app/models/make.rb b/app/models/make.rb
index f76801a..91b1f96 100644
--- a/app/models/make.rb
+++ b/app/models/make.rb
```

```
@@ -1,3 +1,5 @@
 class Make < ApplicationRecord
-  has_many :cars
+  has_many :cars, dependent: :nullify # don't destroy cars when their make is destroyed
+  validates :name, presence: true, uniqueness: true # validate make
+  validates :country, presence: true # validate country
 end
diff --git a/app/models/part.rb b/app/models/part.rb
index 98a4ae0..e1b0288 100644
--- a/app/models/part.rb
+++ b/app/models/part.rb
@@ -1,3 +1,4 @@
 class Part < ApplicationRecord
   has_and_belongs_to_many :cars
+  validates :name, uniqueness: true, presence: true # validate part
 end
diff --git a/app/views/cars/index.html.erb b/app/views/cars/index.html.erb
index b5f40de..abbdc2b 100644
--- a/app/views/cars/index.html.erb
+++ b/app/views/cars/index.html.erb
@@ -2,6 +2,15 @@

 <h1>Cars</h1>

+<!-- Search form for cars -->
+<%= form_tag(search_cars_url, method: "get") do %>
+  <%= label_tag(:query, "Search cars by model or vin:") %>
+  <%= text_field_tag(:query) %>
+  <%= submit_tag("Search") %>
```

```
+<% end %>
+
+<br>
+
 <table>
   <thead>
     <tr>
@@ -15,7 +24,7 @@
   <tbody>
     <% @cars.each do |car| %>
       <tr>
-        <td><%= car.make.name %></td>
+        <td><%= car.make.name if car.make %></td><!-- fix for failing get index url test -->
         <td><%= car.model %></td>
         <td><%= car.vin %></td>
         <td><%= link_to 'Show', car %></td>
diff --git a/app/views/cars/show.html.erb b/app/views/cars/show.html.erb
index 4ea3984..5c0b094 100644
--- a/app/views/cars/show.html.erb
+++ b/app/views/cars/show.html.erb
@@ -2,7 +2,7 @@

 <p>
   <strong>Make:</strong>
-  <%= @car.make.name %>
+  <%= @car.make.name if @car.make %>
 </p>

 <p>
```

```
diff --git a/app/views/layouts/application.html.erb b/app/views/layouts/application.html.erb
index c08d4f7..54d1682 100644
--- a/app/views/layouts/application.html.erb
+++ b/app/views/layouts/application.html.erb
@@ -1,7 +1,7 @@
 <!DOCTYPE html>
 <html>
   <head>
-    <title>CS371001</title>
+    <title>CS-3710-02</title>
     <%= csrf_meta_tags %>
     <%= csp_meta_tag %>

diff --git a/app/views/makes/index.html.erb b/app/views/makes/index.html.erb
index f675780..a82d043 100644
--- a/app/views/makes/index.html.erb
+++ b/app/views/makes/index.html.erb
@@ -2,6 +2,15 @@

 <h1>Makes</h1>

+<!-- Search form for makes -->
+<%= form_tag(search_makes_url, method: "get") do %>
+  <%= label_tag(:query, "Search makes by name:") %>
+  <%= text_field_tag(:query) %>
+  <%= submit_tag("Search") %>
+<% end %>
+
+<br>
```

```
+
 <table>
   <thead>
     <tr>
diff --git a/app/views/parts/index.html.erb b/app/views/parts/index.html.erb
index bbc1cf9..d925d90 100644
--- a/app/views/parts/index.html.erb
+++ b/app/views/parts/index.html.erb
@@ -2,6 +2,15 @@

 <h1>Parts</h1>

+<!-- Search form for parts -->
+<%= form_tag(search_parts_url, method: "get") do %>
+    <%= label_tag(:query, "Search parts by name:") %>
+    <%= text_field_tag(:query) %>
+    <%= submit_tag("Search") %>
+<% end %>
+
+<br>
+
 <table>
   <thead>
     <tr>
@@ -24,4 +33,4 @@

 <br>

-<%= link_to 'New Part', new_part_path %>
```

```
+<%= link_to 'New Part', new_part_path %>
\ No newline at end of file
diff --git a/config/routes.rb b/config/routes.rb
index e15bf14..c57d4e8 100644
--- a/config/routes.rb
+++ b/config/routes.rb
@@ -1,7 +1,22 @@
 Rails.application.routes.draw do
   resources :cars_parts
-  resources :cars
-  resources :makes
-  resources :parts
+  # route for cars search
+  resources :cars do
+    collection do
+      get 'search'
+    end
+  end
+  # route for makes search
+  resources :makes do
+    collection do
+      get 'search'
+    end
+  end
+  # route for parts search
+  resources :parts do
+    collection do
+      get 'search'
+    end
```

```
+  end
   # For details on the DSL available within this file, see http://guides.rubyonrails.org/routing.html
 end
diff --git a/test/application_system_test_case.rb b/test/application_system_test_case.rb
index d19212a..5ee5aae 100644
--- a/test/application_system_test_case.rb
+++ b/test/application_system_test_case.rb
@@ -1,5 +1,5 @@
 require "test_helper"

 class ApplicationSystemTestCase < ActionDispatch::SystemTestCase
-  driven_by :selenium, using: :chrome, screen_size: [1400, 1400]
+  driven_by :selenium, using: :chrome, screen_size: [1360, 768] # modified to be the resolution I typically use
 end
diff --git a/test/controllers/cars_controller_test.rb b/test/controllers/cars_controller_test.rb
index c2f842f..af102ec 100644
--- a/test/controllers/cars_controller_test.rb
+++ b/test/controllers/cars_controller_test.rb
@@ -15,14 +15,65 @@ class CarsControllerTest < ActionDispatch::IntegrationTest
     assert_response :success
   end

+  # Modified to create new car
   test "should create car" do
     assert_difference('Car.count') do
-      post cars_url, params: { car: { make_id: @car.make_id, model: @car.model, vin: @car.vin } }
+      post cars_url, params: { car: { make_id: makes(:one).id, model: cars(:one).model, vin: "ABCD1234", part_ids: [parts(:one).id] } }
```

```
    end

    assert_redirected_to car_url(Car.last)
  end

+  # search shouldn't find missing model
+  test "shouldn't find missing model" do
+    assert Car.where("model like ?", "Turnip").length == 0 # .length is the number of results returned
+  end
+
+  # search shouldn't find missing vin
+  test "shouldn't find missing vin" do
+    assert Car.where("vin like ?", "QQQ666").length == 0 # .length is the number of results returned
+  end
+
+  # search should find model fixture
+  test "should find model from fixture" do
+    assert Car.where("model like ?", "DeLorean").length == 1 # .length is the number of results returned
+  end
+
+  # search should find vin fixture
+  test "should find vin from fixture" do
+    assert Car.where("vin like ?", "ABC123").length == 1 # .length is the number of results returned
+  end
+
+  # search always returns 200 OK
+  test "searches always return 200" do
+    get search_cars_url, params: {query: "T-Rex"}
+    assert_equal 200, status
```

```
+  end
+
+  # should find DeLorean (model)
+  test "should find DeLorean" do
+    get search_cars_url, params: {query: "DeLorean"}
+    assert_select 'td', 'DeLorean'
+  end
+
+  # should find ABC123 (vin)
+  test "should find ABC123" do
+    get search_cars_url, params: {query: "ABC123"}
+    assert_select 'td', 'ABC123'
+  end
+
+  # shouldn't find Porg (model)
+  test "shouldn't find Porg" do
+    get search_cars_url, params: {query: "Porg"}
+    assert_select 'td', false
+  end
+
+  # shouldn't find QQQ666 (vin)
+  test "shouldn't find QQQ666" do
+    get search_cars_url, params: {query: "QQQ666"}
+    assert_select 'td', false
+  end
+
   test "should show car" do
     get car_url(@car)
     assert_response :success
```

```
@@ -33,8 +84,9 @@ class CarsControllerTest < ActionDispatch::IntegrationTest
    assert_response :success
  end

+  # modify update car to test updating of car
  test "should update car" do
-    patch car_url(@car), params: { car: { make_id: @car.make_id, model: @car.model, vin: @car.vin } }
+    patch car_url(@car), params: { car: { make_id: makes(:one).id, model: "DeLorean 2", vin: "A1B2", part_ids: parts(:one).id } }
    assert_redirected_to car_url(@car)
  end

diff --git a/test/controllers/makes_controller_test.rb b/test/controllers/makes_controller_test.rb
index 52dfd39..264c1e5 100644
--- a/test/controllers/makes_controller_test.rb
+++ b/test/controllers/makes_controller_test.rb
@@ -15,14 +15,43 @@ class MakesControllerTest < ActionDispatch::IntegrationTest
    assert_response :success
  end

+  # modified to create new make
  test "should create make" do
    assert_difference('Make.count') do
-      post makes_url, params: { make: { country: @make.country, name: @make.name } }
+      post makes_url, params: { make: { country: "Germany", name: "Volkswagen" } }
    end

    assert_redirected_to make_url(Make.last)
  end
```

```
+  # search shouldn't find missing make
+  test "shouldn't find missing make" do
+    assert Make.where("name like ?", "Kia").length == 0 # .length is the number of results returned
+  end
+
+  # search should find make fixture
+  test "should find make from fixture" do
+    assert Make.where("name like ?", "Porsche").length == 1 # .length is the number of results returned
+  end
+
+  # search always returns 200 OK
+  test "searches always return 200" do
+    get search_makes_url, params: {query: "Tuba"}
+    assert_equal 200, status
+  end
+
+  # should find DMC
+  test "should find DMC" do
+    get search_makes_url, params: {query: "DMC"}
+    assert_select 'td', 'DMC'
+  end
+
+  # shouldn't find Kia
+  test "shouldn't find Warp Core" do
+    get search_makes_url, params: {query: "Kia"}
+    assert_select 'td', false
+  end
+
```

```
  test "should show make" do
    get make_url(@make)
    assert_response :success
diff --git a/test/controllers/parts_controller_test.rb b/test/controllers/parts_controller_test.rb
index 0191216..9760fbb 100644
--- a/test/controllers/parts_controller_test.rb
+++ b/test/controllers/parts_controller_test.rb
@@ -15,14 +15,43 @@ class PartsControllerTest < ActionDispatch::IntegrationTest
    assert_response :success
  end

+  # Modified to create new part
  test "should create part" do
    assert_difference('Part.count') do
-      post parts_url, params: { part: { name: @part.name } }
+      post parts_url, params: { part: { name: "Warp Core" } }
    end

    assert_redirected_to part_url(Part.last)
  end

+  # search shouldn't find missing part
+  test "shouldn't find missing part" do
+    assert Part.where("name like ?", "Trash Panda").length == 0 # .length is the number of results returned
+  end
+
+  # search should find part fixture
+  test "should find part from fixture" do
```

```
+    assert Part.where("name like ?", "Flux Capacitor").length == 1 # .length is the number of results returned
+  end
+
+  # search always returns 200 OK
+  test "searches always return 200" do
+    get search_parts_url, params: {query: "Velociraptor"}
+    assert_equal 200, status
+  end
+
+  # should find Mr. Fusion
+  test "should find Mr. Fusion" do
+    get search_parts_url, params: {query: "Mr. Fusion"}
+    assert_select 'td', 'Mr. Fusion'
+  end
+
+  # shouldn't find Warp Core
+  test "shouldn't find Warp Core" do
+    get search_parts_url, params: {query: "Warp Core"}
+    assert_select 'td', false
+  end
+
   test "should show part" do
     get part_url(@part)
     assert_response :success
diff --git a/test/fixtures/cars.yml b/test/fixtures/cars.yml
index ed8fd7d..83852b5 100644
--- a/test/fixtures/cars.yml
+++ b/test/fixtures/cars.yml
```

```
@@ -1,9 +1,10 @@
 # Read about fixtures at http://api.rubyonrails.org/classes/ActiveRecord/FixtureSet.html

+# created fixture 1
 one:
-  model: MyString
-  vin: MyString
-  make: one
+  model: DeLorean
+  vin: ABC123
+  make: DMC

 two:
   model: MyString
diff --git a/test/fixtures/makes.yml b/test/fixtures/makes.yml
index d8d63f7..067d361 100644
--- a/test/fixtures/makes.yml
+++ b/test/fixtures/makes.yml
@@ -1,9 +1,11 @@
 # Read about fixtures at http://api.rubyonrails.org/classes/ActiveRecord/FixtureSet.html

+# created make one
 one:
-  name: MyString
-  country: MyString
+  name: DMC
+  country: United States

+#created make two
```

```
 two:
-  name: MyString
-  country: MyString
+  name: Porsche
+  country: Germany
diff --git a/test/fixtures/parts.yml b/test/fixtures/parts.yml
index 56066c6..dde6552 100644
--- a/test/fixtures/parts.yml
+++ b/test/fixtures/parts.yml
@@ -1,7 +1,9 @@
 # Read about fixtures at http://api.rubyonrails.org/classes/ActiveRecord/FixtureSet.html

+#created part 1
 one:
-  name: MyString
+  name: Flux Capacitor

+# created part 2
 two:
-  name: MyString
+  name: Mr. Fusion
diff --git a/test/models/car_test.rb b/test/models/car_test.rb
index 39bdaec..7e935f0 100644
--- a/test/models/car_test.rb
+++ b/test/models/car_test.rb
@@ -4,4 +4,67 @@ class CarTest < ActiveSupport::TestCase
   # test "the truth" do
   #   assert true
   # end
```

```
+
+  # test for all empty (presence)
+  test "for all empty" do
+    c = Car.create({:make_id => nil, :model => "", :vin => "", :part_ids => []})
+    refute c.valid?
+    refute c.save
+    assert_equal({:make=>["must exist", "can't be blank"], :model=>["can't be blank"], :vin=>["can't be blank"], :parts=>["can't be blank"]}, c.errors.messages)
+  end
+
+  # test for empty make (presence)
+  test "for empty make" do
+    c = Car.create({:make_id => nil, :model => cars(:one).model, :vin => "A1", :part_ids => [parts(:one).id]})
+    refute c.valid?
+    refute c.save
+    assert_equal({:make=>["must exist", "can't be blank"]}, c.errors.messages)
+  end
+
+  # test for empty model (presence)
+  test "for empty model" do
+    c = Car.create({:make => makes(:one), :model => "", :vin => "B2", :part_ids => [parts(:one).id]})
+    refute c.valid?
+    refute c.save
+    assert_equal({:model=>["can't be blank"]}, c.errors.messages)
+  end
+
+  # test for empty vin (presence)
+  test "for empty vin" do
```

```
+    c = Car.create({:make => makes(:one), :model => cars(:one).model, :vin => "", :part_ids => [parts(:one).id]})
+    refute c.valid?
+    refute c.save
+    assert_equal({:vin=>["can't be blank"]}, c.errors.messages)
+  end
+
+  # test for existing vin (uniqueness)
+  test "for existing vin" do
+    c = Car.create({:make => makes(:one), :model => cars(:one).model, :vin => cars(:one).vin, :part_ids => [parts(:one).id]})
+    refute c.valid?
+    refute c.save
+    assert_equal({:vin=>["has already been taken"]}, c.errors.messages)
+  end
+
+  # test for valid vin (regex)
+  test "for valid vin" do
+    c = Car.create({:make => makes(:one), :model => cars(:one).model, :vin => "C#", :part_ids => [parts(:one).id]})
+    refute c.valid?
+    refute c.save
+    assert_equal({:vin=>["may only contain A-Z and 0-9"]}, c.errors.messages)
+  end
+
+  # test for parts empty (presence)
+  test "for parts empty" do
+    c = Car.create({:make => makes(:one), :model => cars(:one).model, :vin => "C3", :part_ids => []})
+    refute c.valid?
+    refute c.save
```

```
+    assert_equal({:parts=>["can't be blank"]}, c.errors.messages)
+  end
+
+  # test for valid car
+  test "for valid car" do
+    c = Car.create({:make => makes(:one), :model => cars(:one).model, :vin => "D4", :part_ids =>
[parts(:one).id, parts(:two).id]})
+    assert c.valid?
+    assert c.save
+  end
 end
diff --git a/test/models/make_test.rb b/test/models/make_test.rb
index 8162f47..15bf3b4 100644
--- a/test/models/make_test.rb
+++ b/test/models/make_test.rb
@@ -4,4 +4,43 @@ class MakeTest < ActiveSupport::TestCase
   # test "the truth" do
   #   assert true
   # end
+
+  # test for empty name and country (presence)
+  test "for empty name and country" do
+    m = Make.create({:name => "", :country =>""})
+    refute m.valid?
+    refute m.save
+    assert_equal({:name=>["can't be blank"], :country=>["can't be blank"]}, m.errors.messages)
+  end
+
+  # test for empty name (presence)
```

```
+  test "for empty name" do
+    m = Make.create({:name => "", :country => makes(:one).country})
+    refute m.valid?
+    refute m.save
+    assert_equal({:name=>["can't be blank"]}, m.errors.messages)
+  end
+
+  # test for name exists (uniqueness)
+  test "for existing name" do
+    m = Make.create({:name => makes(:one).name, :country => makes(:one).country})
+    refute m.valid?
+    refute m.save
+    assert_equal({:name=>["has already been taken"]}, m.errors.messages)
+  end
+
+  # test for empty country (presence)
+  test "for empty country" do
+    m = Make.create({:name => "Honda", :country => ""})
+    refute m.valid?
+    refute m.save
+    assert_equal({:country=>["can't be blank"]}, m.errors.messages)
+  end
+
+  # test for valid make
+  test "for valid make" do
+    m = Make.create({:name => "Ford", :country => "Pinto"})
+    assert m.valid?
+    assert m.save
+  end
```

```
 end
diff --git a/test/models/part_test.rb b/test/models/part_test.rb
index bba3103..c588061 100644
--- a/test/models/part_test.rb
+++ b/test/models/part_test.rb
@@ -4,4 +4,27 @@ class PartTest < ActiveSupport::TestCase
   # test "the truth" do
   #   assert true
   # end
+
+  # test for empty part name (presence)
+  test "for empty name" do
+    p = Part.create({:name => ""})
+    refute p.valid?
+    refute p.save
+    assert_equal({:name=>["can't be blank"]}, p.errors.messages)
+  end
+
+  # test for part already exists (uniqueness)
+  test "for existing name" do
+    p = Part.create({:name => parts(:one).name})
+    refute p.valid?
+    refute p.save
+    assert_equal({:name=>["has already been taken"]}, p.errors.messages)
+  end
+
+  # test for valid part
+  test "for valid part" do
+    p = Part.create({:name => "Warp Core"})
```

```
+    assert p.valid?
+    assert p.save
+  end
 end
diff --git a/test/system/cars_test.rb b/test/system/cars_test.rb
index 2c1e3f4..d60cd7e 100644
--- a/test/system/cars_test.rb
+++ b/test/system/cars_test.rb
@@ -8,15 +8,51 @@ class CarsTest < ApplicationSystemTestCase
   test "visiting the index" do
     visit cars_url
     assert_selector "h1", text: "Cars"
+    assert_selector "td", text: cars(:one).make # added to test presence of make in table
+    assert_selector "td", text: cars(:one).model # added to test presence of model in table
+    assert_selector "td", text: cars(:one).vin # added to test presence of vin in table
+  end
+
+  # test missing model in search
+  test "missing model in search" do
+    visit cars_url
+    fill_in "query", with: "Cucumber"
+    click_on "Search"
+    refute_selector "td"
+  end
+
+  # test found model in search
+  test "found model in search" do
+    visit cars_url
+    fill_in "query", with: "DeLorean"
```

```
+    click_on "Search"
+    assert_selector "td", text: "DeLorean"
+  end
+
+  # test missing vin in search
+  test "missing vin in search" do
+    visit cars_url
+    fill_in "query", with: "QQQ666"
+    click_on "Search"
+    refute_selector "td"
+  end
+
+  # test found vin in search
+  test "found vin in search" do
+    visit cars_url
+    fill_in "query", with: "ABC123"
+    click_on "Search"
+    assert_selector "td", text: "ABC123"
   end

   test "creating a Car" do
     visit cars_url
     click_on "New Car"

-    fill_in "Make", with: @car.make_id
+    select "DMC", :from => "car_make_id" # modified to test dropdown
     fill_in "Model", with: @car.model
-    fill_in "Vin", with: @car.vin
+    fill_in "Vin", with: "YYY" # modified to update to new vin because vin must be unique
```

```
+    check "Mr. Fusion" # added to update because validation requires parts presence
     click_on "Create Car"

     assert_text "Car was successfully created"
@@ -27,9 +63,9 @@ class CarsTest < ApplicationSystemTestCase
     visit cars_url
     click_on "Edit", match: :first

-    fill_in "Make", with: @car.make_id
+    select "DMC", :from => "car_make_id" # modified to test dropdown
     fill_in "Model", with: @car.model
-    fill_in "Vin", with: @car.vin
+    fill_in "Vin", with: "ZZZ" # modified to update to new vin because vin must be unique
     click_on "Update Car"

     assert_text "Car was successfully updated"
diff --git a/test/system/makes_test.rb b/test/system/makes_test.rb
index 5751726..52ffca3 100644
--- a/test/system/makes_test.rb
+++ b/test/system/makes_test.rb
@@ -8,6 +8,24 @@ class MakesTest < ApplicationSystemTestCase
     visit makes_url
     assert_selector "h1", text: "Makes"
+    assert_selector "td", text: makes(:one).name # added to test presence of name in table
+    assert_selector "td", text: makes(:one).country # added to test presence of country in table
+  end
+
+  # test missing make in search
```

```
+  test "missing make in search" do
+    visit makes_url
+    fill_in "query", with: "Kia"
+    click_on "Search"
+    refute_selector "td"
+  end
+
+  # test found make in search
+  test "found make in search" do
+    visit makes_url
+    fill_in "query", with: "DMC"
+    click_on "Search"
+    assert_selector "td", text: "DMC"
   end


  test "creating a Make" do
@@ -15,7 +33,7 @@ class MakesTest < ApplicationSystemTestCase
    click_on "New Make"


    fill_in "Country", with: @make.country
-   fill_in "Name", with: @make.name
+   fill_in "Name", with: "Toyota" # modified to update part name because validation requires unique name
    click_on "Create Make"


    assert_text "Make was successfully created"
@@ -27,7 +45,7 @@ class MakesTest < ApplicationSystemTestCase
    click_on "Edit", match: :first
```

```
    fill_in "Country", with: @make.country
-   fill_in "Name", with: @make.name
+    fill_in "Name", with: "Kia" # modified to update make name because validation requires unique name
    click_on "Update Make"

    assert_text "Make was successfully updated"
diff --git a/test/system/parts_test.rb b/test/system/parts_test.rb
index 7bdf358..934eb04 100644
--- a/test/system/parts_test.rb
+++ b/test/system/parts_test.rb
@@ -8,13 +8,30 @@ class PartsTest < ApplicationSystemTestCase
   test "visiting the index" do
     visit parts_url
     assert_selector "h1", text: "Parts"
+    assert_selector "td", text: parts(:one).name # added to test presence of part in table
+  end
+
+  # test missing part in search
+  test "missing part in search" do
+    visit parts_url
+    fill_in "query", with: "Velociraptor"
+    click_on "Search"
+    refute_selector "td"
+  end
+
+  # test found part in search
+  test "found part in search" do
+    visit parts_url
```

```
+    fill_in "query", with: "Flux Capacitor"
+    click_on "Search"
+    assert_selector "td", text: "Flux Capacitor"
   end

   test "creating a Part" do
     visit parts_url
     click_on "New Part"

-    fill_in "Name", with: @part.name
+    fill_in "Name", with: "New Part Name" # modified to update part name because validation requires unique name
     click_on "Create Part"

     assert_text "Part was successfully created"
@@ -25,7 +42,7 @@ class PartsTest < ApplicationSystemTestCase
     visit parts_url
     click_on "Edit", match: :first

-    fill_in "Name", with: @part.name
+    fill_in "Name", with: "Updated Part Name" # modified to update part name because validation requires unique name
     click_on "Update Part"

     assert_text "Part was successfully updated"
```