Method 1:

1. Configure config/database.yml
   Put in your new database: for example:

```
# SQLite version 3.x
#   gem install sqlite3
#
#   Ensure the SQLite 3 gem is defined in your Gemfile
#   gem 'sqlite3'
development:
  adapter: sqlite3
  database: db/development.sqlite3
  pool: 5
  timeout: 5000


apeers:
  adapter: sqlite3
  database: ../../1/job14/apeers/db/development.sqlite3
  pool: 5
  timeout: 5000

# Warning: The database defined as "test" will be erased and
# re-generated from your development database when you run "rake".
# Do not set this db to the same as development or production.
test: &test
  adapter: sqlite3
  database: db/test.sqlite3
  pool: 5
  timeout: 5000

production:
  adapter: sqlite3
  database: db/production.sqlite3
  pool: 5
  timeout: 5000

cucumber:
  <<: *test
```

Apeers is the old rails application, I want to use the database from “apeers” in my new rails application.

After you configure the old database, you can create a model without migration:

For example, in old application there is model named: “product”

Then in my new application, you have to create a model with exactly the same name, and you cannot migrate, as migrate is for add/ delete/change something in the database.

In new app: rails g model product –migration=false

In the new product model:

```
Class Product < ActiveRecord::Base

   establish_connnection: products

end
```

Then all have done, you could use the old database of “apeers” now in the new rails application.

If you want to use lots of model from old application, I highly recommended you to difine a parent class for all tables , for example:

```ruby
class External < ActiveRecord::Base
  self.abstract_class=true
  establish_connection :apeers
end
```

```ruby
class Product < External

end
```

Method 2: using ActiveResourses

The old rails app located in localhost:3000 web port.

```ruby
class Showcase < ActiveResource::Base
  self.site = "http://localhost:3000"
end
```

Firstly you have to change the new rails app in a new web port:

For example: in config/boot.rb file: add the codes(you can change 3003 to the new port you want)

```ruby
module Rails
  class Server
    alias :default_options_alias :default_options
    def default_options
      default_options_alias.merge!(:Port => 3003)
    end
  end
end
```

Then in your model :

```ruby
class Product < ActiveResource::Base
  self.site = "http://localhost:3000"
end
```

Then you can also access the database from odd application now.

These two methods could help you access the database, but if you want to share the uploaded images or pdf, then you could use symbolic link, to share the folder which keeps the uploaded files.

For example, share the products folder between these two apps.

```
C:\Windows\system32>mklink /J C:\Users\dsun\Documents\1\adetails\public\products
   C:\Users\dsun\Documents\1\job14\apeers\public\system\products
```