- There are many different model validations. They are performed on the field level. Some examples are:
  - Validate the presence of a field
  - Validate that a field is a number
  - Validate that a field is formatted like an email address
  - Validate that a field is unique (like a username)
- Implement validations through the use of a straightforward DSL in the model class:

```
class Movie < ActiveRecord::Base

  validates :name, presence: true

end
```

```
class User < ActiveRecord::Base

  validates :username, presence: true, uniqueness: true

end
```

- An invalid model object won't be able to be saved to the database

```
@movie = Movie.new
@movie.name = ""
@movie.save
=> false
```

- You should never store user passwords in plain text.
- Instead, use a one-way hash that's virtually unbreakable: the **bcrypt** algorithm**.**
- Use the **session** hash to store browser data in cookies and maintain state.
- Use a **sessions_controller** to manage login and logout
- Implement BCrypt in Rails using **has_secure_password**
    1. Add the bcrypt gem to your **Gemfile**
    2. Add a line that reads **has_secure_password** to your User model
    3. Add a field called **password_digest** on your User model to hold the encrypted password data
    4. When creating the user, set the plain text password in the **password** attribute