

BAZE I MODELI

\$ rake db:create - pravi bazu (ne osnov config/database.yml podataka)

\$ rails generate migration → stvara tablicu

\$ rails generate model User → automatski radi i migraciju



rails generate model User name:string email:string
t.string "first_name", :limit => 25

RUN-anje migracija

\$ rake db:migrate {VERSION = x} (\$rake db:seed)

1. GENERIRAT MODELE (JEDNINA) (CAMEL-CASE)

2. NAPISAT MIGRACIJE - FOREIGN KEYS - INDEX

3. RUN-AT MIGRACIJE

- MIGRATION METODE
 - add-column (table, column, type, options)
 - remove-column (table, column)
 - rename-column (table, column, new-name)
 - change-column (table, column, type, options)
 - add-index (table, column, options)

- DA BI SPRJEĆIO MASS
ASSIGNMENT VULNERABILITY U:
config/application.rb komentirati
↳ config.active_record.whitelist_attributes = true

- SKO SVAKI MODEL MORA EKSPlicitno
IMAT attr_accessible :property za
ONE PROPERTYE KOJIMA JE DOZVOLJEN
MASS ASSIGNMENT / update_attributes
METODA

→ mijek na foreign keyove

Model & relations

MODEL

NEW / SAVE TEHNika → OVO KAO SHT.

HASH

SUBJECT = Subject.new (:name => "Ime", :position => 1, :visible => true)
subject.save ↗ TRUE
↘ FALSE

CREATE TEHNika - automatski sprema

HASH

subject = Subject.create (...)

UPDATE

subject = Subject.find(1) ↗ ID

HASH

↗ TRUE
↘ FALSE

subject.update_attributes(...)

DELETE

subject = Subject.find(x)

subject.destroy

FIND

Subject.find_by_id(x) ili Subject.find_by_name("Ime")

Subject.all

subject.first ili subject.last

QUERY

Subject.where(:visible => true).order("position asc")
↘ MORE SE CHAINAT
↙ VRAĆA ARRAY

ORDER, LIMIT, OFFSET

- order ("sql")
- limit (integer)
- offset (integer) ^{SQL}

Subject.order("position ASC").limit(20).offset(40)

table-name.column-name ASC/DESC

"subjects.name DESC"

"subjects.visible DESC, subjects.name ASC"

NAMED SCOPE

→ u MODELU SU STOREANI

- scope :ime, WHERE(...)

- subject = Subject.ime

-

SEARCH

- scope :search, lambda { |query| where(["name LIKE ?","%" + query + "%"])}
? ili

? scope :sorted, order ("last_name ASC", "first_name ASC")

RELACIJE U TABLICAMA

- KLAST se "belongs_to" ima foreign key

- nijek definisat objektnu relaciju

↳ u MODELIMA

belongs_to :subject

ONE-TO-MANY

↳ VRIĆA ARRAY OBJEKATA (UMJESTO 1. OBJEKTA)

PLURAL

has-many :pages

→ subject.pages → sve stranice (array)

- subject.pages << page (dodava stranicu)
- subject.pages.delete (page)
- subject.pages.clear → čisti vešnice
- subject.pages.empty?
- subject.pages.size → koliko je stranica?

MANY-TO-MANY - LYNAT & VIA MANY-TO-MANY
 → POTREBAN JOIN TABLE
 → JOIN TABLE NEKA primary key (:id => false)

Page has_and_belongs_to_many :admin_users → obj stranice
 stavit vseb

JOIN TABLE IME

↳ obje tablice v pluralu

↳ ALFABETSKI RED

OPS. JOIN TABLICE

→ FILE

SIMPLE-CMS

↳ DB

↳ MIGRATION

↳ CREATE ADMINUSERS...

rails generate migration CreateAdminUsersPagesJoin

~~RICKY MANY-TO-MANY~~

~~-SIMPLE-KEYS → SECTIONALIT~~ ~~MODEZ i obveznosti~~

• u migraciji

↳ :id => false

↳ istemo foreign keyevi

↳ les timestampova

↳ add_index :admin_users_pages, ["admin_user_id", "page_id"]

OVAKAV INDEX TREBA 0004T ^{ZAJEDNO}
 NJE GREKA

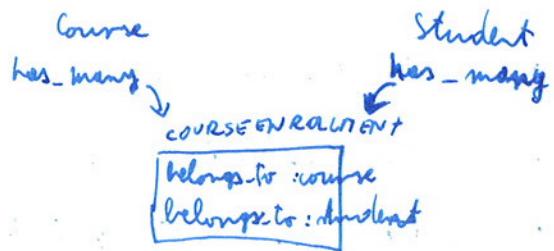
• u modelu:

has_and_belongs_to_many :editors, :class_name => "AdminUser"

DEUKČIJE IME
 KOJE JE ZAPRATO OVO

MANY TO MANY RICH

- one join table rovemo kako hocimo
- join tablica ima:
 - ↳ :id => imao pa
 - ↳ dva foreign keys
 - ↳ ima svoj MODEL



① napravit model za join table

↳ dodat foreign keyeve

↳ dodat "drugaciji" index: add_index :section_edits, [:admin_user_id], 'sectionid'

↳ imi timestamps

② MODELI

→ u "glavne tablice dodat" JOIN TABLE

↳ has_many :section_edits

↳ has_many :sections, :through

↳ DRUGA GLAVNA TABLICA SE SAD MOZE "DODAVATI" NPR. page_sections

JOIN TABLE

↓

→ u join tablici dodat

2x → belongs_to :editor, :class_name => "Admin User", :foreign_key => 'admin_user_id'

belongs_to :section

DRUGO IME

PRVAKLASA

TREBA DODAT BOJ

↓ JE ID

Controller & crud

CONTROLLER & CRUD

• TIPIČNI CRUD

↳ jedan controller po modelu

↳ ime u množini (Subjects Controllers)

\$ rails generate controller Subjects

PLURAL
↓
new show ...
↑ ↑
actions in controller



READ LIST

- u controller dodajemo činio što dohvaćamo iz modela + @variables
 - def list
 - @subjects = Subject.order('subject.position ASC')
- u view dohvaćamo @instance var. => **list.html.erb**
- ↳ = link_to ("Show", { :action => **'show'**, :id => subject.id }, :class => action("show")) %>

SHOW

- za ovu treba užeti objekt imati :id
- u controlleru
 - def show
 - @subject = Subject.find(params[:id])
- u viewu **show.html.erb** → pozivat ~~je~~ stvara od @subject

GOOD TO DO ZA READ

```
def index
  ↗ AKO SE NE UPISU SUBJECTS / LIST NEGOTIATION SUBJECTS
  list ← VRŠI LIST
  render('list') ← I PRIMAKI LIST.HTML.ERB (A NE INDEX.HTML.ERB)
end
```

FORME

```

<% = form_for(:subject, :url => { :action => 'create' }) do |f| %>
<% = f.text_field(:name) %>
<% = f.text_field(:position) %>
    ...
<% = submit_tag("Create Subject") %>
<% end %>

ili

```

```

<% = form_tag(:action => 'create') do %>
<% = text_field(:subject, :name) %>
<% = text_field(:subject, :position) %>
    ...
<% = submit_tag("Create Subject") %>
<% end %>

```

CREATE new - powrzuje formu

- VIEW → NEW.HTML.ERB
 - ↳ elementy formy i modelu

• CONTROLLER

→ NE TREBA NIŠTA
 ili?
 Boże

→ DEF new

① subject = Subject.new
 and

↳ u formy dodaje pre-definiowane
 elementy (z inicjalami)

↳ ili Subject.new(:name => 'default')

- JEDNO
 SAD
 PREDEFINIOWANE

ACTION CREATE - PROCESIRAT FORMU

• CONTROLLER

```
def create
```

```
@subject = Subject.new(params[:subject]) → INSTANCIROVANJE NOVI  
OBJEKTA
```

```
if @subject.valid?
```

→ OVAJ SE SPRETA
+ ISPITIVA ŠTA JE
AKO JE OK ISPISIVA SVE

```
redirect_to(:action => 'list')
```

→ AKO FORMA NIE JE OK POPUNJEN
VRACA NA FORMU KOJA JE
PRE-POPULIRANA

```
else
```

```
render('new')
```

```
end
```

```
end
```

• PODAT LINKA ZA NEW SUBJECT → U LIST.HTML.ERB

```
<% = link_to("Add new", { :action => 'new' }, :class => 'action new') %>
```

UPDATE edit - display form

↳ ZAHTEVA ID OBJEKTA DA JE MOže UPDATEAT

↳ UPDATE ACTION KORISTI FIND I UPDATE ATTRIBUTES
↓ OVO U LIST.HTML.ERB

```
<% = link_to("Edit", { :action => 'edit', :id => subject.id }, :class => 'action edit') %>
```

• CONTROLLER edit

```
↳ def edit → ini kev show
```

• EDIT.HTML.ERB

↳ ISTI KAO NEW.HTML.ERB

↳ KOZMETICKE PROMJENE NEW ⇒ UPDATE ISL

↳ PROMJENIT { :action => 'update', :id => @subject.id } ↳ DIMNO?

• CONTROLLER update

```
↳ @subject = Subject.find(params[:id])
```

```
if @subject.update_attributes(params[:subject])
```

```
redirect_to(:action => 'show', :id => @subject.id)
```

```
else
```

```
render('edit')
```

```
end
```

→ PRIJEUJE
OBJECT

DELETE

DELETE - displays form (form ne more ni postojat)
 KUJE
 DESTROY - procesirava.

↳ unijk težnji :fd
 ↳ destroy knisti find i destroy metode
 Delete page može bit po idom ↳ uporavje
 ↳ popis atributa
 ↳ konfirmacija

n LIST.HTML.ERB

- <%= link_to("Delete", { :action => 'delete' }, { :id => subject.id }, { :class => 'action delete' }) %>
- delete.html.erb → po želi forme
 ↳ forma mora imati :id => @subject.id
- CONTROLLER
 - ↳ def delete → iti kao edit
 - ↳ def destroy
 - Subject.find(params[:id]).destroy
 - redirect_to (:action => 'list')

end

HASH FLASH

flash[:notice] = "Subject created successfully"

- ↳ dodat ponike prije redirect_to... n controllerima za...
- ↳ def create
- ↳ def update
- ↳ def destroy
- n list.html.erb (iši kdo kriji dungi template)


```
<% if !flash[:notice].blank? %>
<div class="notice">
  <%= flash[:notice] %>
</div>
<% end %>
```

VIEW TEMPLATE

↳ ovaj je dodaje u templateu

LAYOUTS

```

<html>
  <head>
    </head>
    <body>
      <%= yield %>
    </body>
  </html>

```

u kontroleme:

layout 'admin' → kreira layout admin.html.erb

- u layoutima možemo stvoriti instance variable koje su
* korišćenje podjeli yield dokumentu
- u layout stvara se flash var

PARTIALS - partial templates

- koristi se da dionicim html.erb-ovim tipa new i edit
- ime im uvek ima -prefiks = -form.html.erb
- u dokumentima za -form poziva se:


```

<%= render(:partial => "form", :locals => { :f => f }) %>
      
```

BETW CTRF

variable u dokumentu trenutnom
variable u -form

STYLE SHEETS

- ↳ nazore se u /public/stylesheets
 - ↳ ekstenzija .css
- CSS dodavamo u file views/layouts (u head tag)
 - ↳ <%= stylesheet_link_tag('public', :media => 'all') %>
 - ↳ in prvoje -za drugi CSS
'admin',

JAVA SCRIPTS

- ↳ lociramo /public/scripts
- ↳ razvijavaju se na JS
- ↳ po defaultu skripte u application.js
- u reportu (views/layouts) dodat


```
<%= javascript_include_tag('application') %>
          , 'drugi'
```
- POZIVANJE JS preko LINKA


```
<a href="#"= link_to('Roar', '#', :onclick => "js Roar(); return false;" ) %>
```
- POZIVANJE JS - ODRZAVI


```
<%= javascript_tag("js Roar();") %>
```
- Izvršavanje JS-a


```
<%= javascript_tag("alert('Hello.');" ) %>
```
- Izvršavanje JS-a INPUTANOG od usera


```
<%= javascript_tag("alert('#{escape_javascript(text)}');") %>
```

NEKA VARIJABLA
ODV. OSIGURAVA DA JE TEXT
NJE MALICIOZNI JAVASCRIPT KOD

USER
ILI INPUT

IMAGES

- ↳ /public/images
- ↳ nova defaultna file extenzije
- ↳ može biti subfoderi
- slike


```
rails.png
```

 $\langle\% = image_tag('rails.png') \%\rangle$
- opisje


```
alt text
```

 $\langle\% = image_tag('rails.png', :size => 150x60, :alt => "Rails logo") \%\rangle$

velicina u pix

Text helpers

TEXT HELPERS

- <% = word_wrap(text, :line_width => 30) %>
variable na textom \rightarrow organizirava Širinu retke ne 30 mukova
text = "text..."
- <% = simple_format(text) %> \rightarrow prehvata \n \rightarrow
 tag radi html-a
 \rightarrow 2x \n (\n\n) je novi paragraf
- <% = truncate(text, :length => 40, :omission => "...") %>
PRVIH 40 ZNAKOVA. OSTALO ZAMJENI S OVIM
- <% = excerpt(text, 'rijec', :medium => 7, :omission => "...") %>
OVO TEZI I PRIKAZUJE 7 ZNAKOVA OVO JE RIJEC
NADE U TEXTU OSTALO ZAMJENI S OVIM
- <% = highlight(text, 'text example') %>
ili
<% = highlight(text, ['example', 'ramble'], :highlighter => '*1* *2*') %>
NIZ RIJECI HIGHLIGHTA OVAKO HIGHLIGHTA
- OVO INACI PRVA RIJEC U NRU V OVRU PRIMJERU JE TO 'EXAMPLE'
- <% = auto_link(text) %> - linkove u textu automatski prehvata u "klikabilne" linkove
- APOMENA
AT BI AUTO_LINK RADIO - STAVIT TEXT = "...", HTML_SAFE
DA RAILS ZNA DA JE VARIJABLA OK
- pluralize(n, 'octopus') \rightarrow pomeni pluralize

NUMBER HELPER

Number & date helperi

IZNOV

DECIMALNIH
↓

JEDINICA

JEDINICA
VALUTE

PREVOBRY

number-to-currency (34.5, :precision => 0, :unit => "kr", :format => "%n %")
→ 35 kr

DATE & TIME HELPERI

second	seconds
minute	minutes
hour	:
day	:
week	ČETVrtak
month	MNOŽINA
year	ISTA STVAR

30.days ⇒ 30 dana u sekundama
id

Time.now

Time.now + 30.days - 23.minutes

NOTE SE ZBRAJAT (IVE JE U SEKUNDAMA)

- ago
- from_now
- npr. 30.days.from_now
- beginning-of-day
- -week
- -month
- -year
- yesterday/tomorrow
- end-of-day
- prev-month/next-month
- -week
- prev-year/next-year
- -month
- -year
- chainable je ne! Npr. Time.now.prev-year.beginning-of-month

• strftime ⇒ string from time

Time.now.strftime ("%B %d, %Y %H:%M")

MJESEC

SAT

MINUTA

• Time.now.to_s(:long) :time "13:36"

ili

{ Time::DATE_FORMATS[:custom] = "%B %e, %Y at %I:%M %P"

Time.now.to_s(:custom)

UZIMA OVAJ
FORMATING

OVU LINIJU STAVIT U FILE: /config/initIALIZERS/date-formats.rb

NAPRAVIT NOVI FILE

CUSTOM HELPERS

/app/helpers

↳ tu se naloži helperi za modele kjer smo reči nepraktični

↳ kad smo u npr. Subjects controllerju → mora se izvajati helper metoda iz Subjects helpers (ali NE iz drugih helpers)

↳ isto i za views

↳ application-helper je globalen (za vse)

STATUS TAG → POKLEDNIK V SIMPLE-CMS APP

FORMFORM HELPERS

- form_for
- text_field
- password_field
- text_area
- hidden_field
- file_field
- radio_button
- check_box

OVO JE MORA
BIT AKO IMATI UPLOAD BUTTON

form_for(:subject, :url => { :action => "create" },
 :html => { :multipart => true }) do |f|
 f.text_field(:name, :size => 40, :maxlength => 40)
 f.text_field(:subject, :alt_name, :size => 40, :maxlength => 40)
 f.password_field(:password, :size => 40, :maxlength => 40)

OVO JE ISTA SIVAR KAO NOGAO JE BIT I
REFE PRIJE NEKI DRUGI OBJEKAT

f.text_area(:description, :size => "40x5") → DIMENZIJE POLJA

NE VIDI
SE PASSWORD

ZA OGRANICAVANJE UNOSA
TRFBA: JAVASCRIPT

f.hidden_field(:token) → skrivno polje

f.file_field(:logo) → zahtjeva mre :multipart => true

f.radio_button(:content_type, "text")
 f.radio_button(:content_type, "HTML") } ODAZIR IZMEĐU
 f.check_box(:visible) → super je za boolean } NE ZABORAVIT
 submit_tag("Submit") DVE DIVJE OPCIJE STAVIT HTML
 end RVEC IZA KUCICA

HTML radio buttons:

```
// f is a form
<% Category.all.each do |category| %>
<div>
  <%= f.radio_button :category_id, category.id %><%= category.name %>
  <%= f.label :category_id, category.name %>
</div>
<% end %>
```

** html look*

```
<p>
<div>
  <input id="post_category_id_1" name="post[category_id]" type="radio" value="1" />This is category name
  <label for="post_category_id_1">This is category name</label>
</div>
<div>
  <input id="post_category_id_2" name="post[category_id]" type="radio" value="2" />batman
  <label for="post_category_id_2">batman</label>
</div>
</p>
```

FORM SELECT HELPERS

(postaviti izbornik)

f. select (attribute, choices, options = {}, html_options = {})
ili
select (object, attribute, ...)

Opis:

:include_blank => false

:selected => object.attribute

f. select (:position, 1..5, {}, { :style => "width: 4em;" })
OVO TREBA
P BIT TU
1 2 3 4 5
IZBOR OD
A DO 5
HTML

f. select (:content_type, ['text', 'HTML'])
Z OPCIJE U KRUZU
HASH
USERVOD "VISIBLE" I "HIDDEN"
A SPRENAJU SE A IZ

f. select (:visible, { "visible" => 1, "hidden" => 2 })

f. select (:page_id, Page.all.collect { |s| [s.name, s.id] })
OVO JE JEDNA OPCIJA
[name, id], name, id
PRIKAZUJE SE USERU
SUBMITTA SE

DATE & TIME FORM HELPERS

- date_select
- time_select
- datetime_select

naprave nemoresna polja
za unos na postavim izbornicima

FORM TAG HELPERS

(POGLAVLJE 11)

- ↳ redovjeni od active record uslova
- prije je bilo form_for nad → form_tag
 - ↳ i takvo se može elemente od forme
- mogu se mixat sa form_for
 - select_tag ("gender", options_for_select(["male", "female"], selected_text))

KOJA VRIJEDNOST
JE PRE-SELEKTIRANA

FORM ERRORS

- validate_presence_of :name

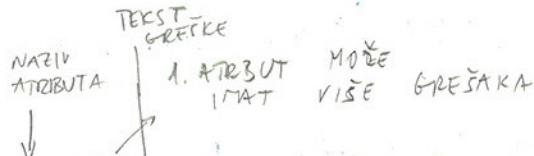
ATRIBUT

IDE U MODEL

- object.errors → uključuje greske u ovo

{
subject
company}

- object.errors.clear
 - :size
 - :each { |attr, msg| ... }
 - .full_messages.each { |msg| ... }



- napraviti vlastiti HTML za greske views/shared/_error-messages.html.erb

↳ OVAJ DIR SAMI NAPRAVIMO
OVAJ FILEIMA VARIABLE KOJE "VUCE"
OSIGURAT DA IH DOBLJE IZ NER
- FORM.HTML.ERB

- u_form.html.erb na svih

Lis = render(:partial => 'shared/error-messages', :locals => { :object => @subject })

RENDERIRAJ NEKI

PARTIAL LAYOUT FILE

↓
LOKALNE
VARIABLE
TU U
FILEU

↑
OVAKO SE
ZONE

↓
OVAKO SE
ZONE U
- FORM.HTML.ERB

U - ERROR-MESSAGES

JOS BOLJE

- STAVIT U APPLICATION HELPER \Rightarrow TU DEFINIRATO POZIVANJE OVOG PARTIAL LAYOUTA


```
def error-messages-for (object)
end
render(:partial => 'shared/error-messages', :locals => {object => object})
```
- U - FORM. HTML. ERB. POZIVATI HELPER

Lg = error-messages-for(@subject) %>

OVDJE JE DEPRECATED
POZIVAT NOVO NA KRAJU
OUT OF BUDGET!

SADA TO LAKO NAPRAVIMO I U
DRUGIM KLASAMA, A PROMJENIMO
SATNO VARIJABLJU

\rightarrow NE ZABORAVIT DOAT views/shared/_error-messages.html.erb
 \rightarrow NE ZABORAVIT DA OVA ATOTEKA IMATI ISPREDO IMENA

FORM FIELD LABELS

- label (object, attribute, text)
- f. label (attribute, text)
- label-tag (name, text).
- pr
 - f. label (:name) \rightarrow u formi da ne moremo print <th> Name </th>
KAO BONUS IME SE ZACRVENI AKO JE GREŠKA P.
 - f. label (:name, "ime") \rightarrow CSS \Rightarrow field-with-errors

ESCAPE INPUT

- ↳ da se na input od korisnika besplatno renderira (da nije SQL injection)
 - cookies
 - form parametri
 - cookie data
 - database data
- ↳ u svaki escapea data su Hush i tney tipa
 - subject. where (:name => user_query, :visible => 1)
 - subject. where ("name=? AND visible=1", user_query)

NOTE BIT I VISE
V PITNIKA, ONDA TREBA
BIT VISE VRIJEDOSTI TU

- neis escape automatski dodavanje atributa
 - subject_update_attributes (:name => "subject 1")

ESCAPING OUTPUT

- METODE
- escape → izbjegavanje izvršavanja JAVASCRIPTA koji je ubačen umjesto varijable ili polja
 - html_escape(), h6
 - raw() → izbjegava escape (html i JS se izvršavaju)
 - html_safe, html-safe? → ispituje da li tekst JS-a i HTML-a
 - u velikini 3 raw je tečet automatski escape
 - ↳ mi trebamo reći velikim da je HTML safe

- strip_links(html) → mije linkove
- strip_tags(html) → mije tagove tipa <p>,
</br>, <ahref...>
- sanitize(html, options) → mije raw ~~raw~~ JS
 - ↳ tags => ['p', 'br', ...], attributes => ['id', 'class', 'style']
 (onda se propusta (WHITE-LISTA) navedeno, a sve ostalo se BLACK-LISTA)
- text = "text... ", html-safe → rad je varijable varijete rate

DATA VALIDATION

Data validation

↳ u modelu?

- validates_presence_of (:name, :option)
 - :message \Rightarrow "can't be blank" \rightarrow more se customizable
- validates_length_of (:name, :option)
 - is, :minimum, :maximum (integer)
 - within, :in (range)
 - :wrong_length \Rightarrow "is the wrong length (should be {{count}} characters)"
 - :too_short \Rightarrow "is too short ...".
 - :too_long \Rightarrow "is too long ..."
- validates_numericality_of (:number, :options, ...)
 - :equal_to, :greater_than, :less_than, :greater_than_or_equal_to,
 - :less_than_or_equal_to (numeric)
 - :odd, :even, :only_integer (boolean)
 - :message \Rightarrow "is not a number"
- validates_inclusion_of
 - :in (enumerable)
 - :message \Rightarrow "is not included in the list"

\rightarrow atribut more let list¹² lists isloxe (array ih range)
- validates_exclusion_of
 - :message \Rightarrow "is reserved"

$\left.\begin{matrix} \text{ISKO KAO} \\ \text{OVO} \end{matrix}\right\}$ ORC
- validates_format_of
 - :with (regular expression)
 - :message \Rightarrow "is invalid"

EMAIL - REGEX

VERIKAJ SLOVAK
OVO DODAT KAO KONSTANTU
U MODELU

$$\text{EMAIL_REGEX} = / ^ [A-Z0-9,_.%+_-] + @ [A-Z0-9,-] + \backslash . [A-Z] \{2,4\} \$ / \text{c}$$

• validates - uniqueness - of

↳ atribut ne može postojati u bazi za taj objekt

:case-sensitive (boolean)

: scope (column symbols for limiting scope)

: message ⇒ "has already been taken"

• validates - acceptance - of "I agree to the services of license."

↳ istvariti virtualni atribut ako nema stupca u atributu

: accept (expected value, "1")

: message ⇒ "must be accepted"

NE SPREM
SE U BAZU

• validates - confirmation - of "emailove ili parolice"

↳ atribut se mora potvrditi uočenjem dva puta

↳ istvariti virtualni atribut npr :email ⇒ :email_confirmation

↳ validirati potvrdu ako atribut nije nil

U FORMAMA SE TO KORISTI KAO

: message ⇒ "doesn't match confirmation" :email_confirmation

OPCIJE za VALIDACIJE

- KE TODE IZVLAČENJU
 - :allow-nil ⇒ true → ne validirati ako atribut ⇒ nil (nije potreban)
 - :allow-blank ⇒ true → isto preškaci ako je atribut (nil, false, "", [], {})
 - :on ⇒ :save / :create / :update
 - ↳ validirati samo ako je nudi o odabranoj akciji
 - :save → validirati u slučaju (default)
 - ~~:if ⇒ :method / :unless ⇒ :method~~
 - ↳ method je metoda definirana u modelu koja vrati true ili false

s = Subject.new

s.errors → vrati hash na greskama

s.valid? → proverava validacije i vrati true ili false

s.errors.full_messages ⇒ ["Name can't be blank"]

PROŠTO
SVE VARIJANTE

NIJE
PODOĆNO
VARIJANTE

IME ATRIBUTA

VALIDATES METHOD

↳ very validation

- intro kao "normalne" validacije, ali brže

- validates :email, :presence => true,
:length => { :maximum => 50 },
:uniqueness => true,
:format => { :with => EMAIL_REGEX },
:confirmation => true

→ trenutno je teško i nikako promjeniti error message...

VALIDATES ASSOCIATED METHOD

- validates_associated (association, options)

↳ asocijacijski objekti moraju biti valid

↳ same very validations

↳ tako neke asocijacije objekta => neće biti greska.

DO NOT PUNE

↳ validates_presence_of
object

? infinite loops & long cascades

- class Auction < ActiveRecord::Base

has_one :car

validates_presence_of :car

validates_associated :car

end

USER AUTHENTICATION

HASHING PASSWORD

↳ ne spremanje parolice kao tekst!

↳ algoritmi: MD5, SHA-1, SHA-2

↳ SHA1?

↳ IRBU

Digest :: SHA1. hexdigest (variable)

- u modelu, uklj, prije class

→ require 'digest/sha1'

→ def self.hash (password = "")

 Digest::SHA1.hexdigest(password)

end

- prije ne:

MORAMO OVO
POZVAT JER SMO TU
DEFINIRALI PASSWORD HASHTEDOU

user.password = admin.user.hash ('password text')

SALTING PASSWORDS

- password + "salt string" => SHA1 => ~~ne~~ nadje password pre-dug za rainbow tables
- salt koji uključi i username => SHA1 za koji treba i pars i username
- random salt → uključi i random broj
- password + Time.now
- ↳ store salt in database => hash it
- unique, random, encrypted salt

Callbacks

MODEL

- user.rb: require 'digest/sha1'
- def self.make_salt(username = "")
Digest::SHA1.hexdigest("Use #{username} with #{Time.now} to make salt")
end
- def self.hash_with_salt(password = "", salt = "")
Digest::SHA1.hexdigest("Put #{salt} on the #{password}")
- Hypothese:
user.salt = AdminUser.make_salt("memorable string ili varijabla")
user.hashed_password = AdminUser.hash_with_salt("password string", user.salt)

OVO JE RANDOM I SPREMIJUO U SVAKOG USERA
 OVO JE SALT SPREMIJUO U BAZU

MASS ASSIGNMENT

- attr_protected
 - ↳ attributi se preklapaju u mass assignmentu
- attr_accessible
 - ↳ definiraju se dostupni → mi stali su restrikciji
- u modelu
 - attr_protected : hashed_password, :salt

→ SAMO U MASS ASSIGNMENTU
 - DIRECT ASSIGNMENT
 - JE ISTI
 User.name => ...

CALLBACKS

- ne se definisu u modelu

	<u>CREATE</u>	<u>UPDATE</u>	<u>DESTROY</u>
PREV VARIABLE	before_validation	before_validation	
POSYNE VARIABLE	after_validation	after_validation	

← VALIDACIJA

PREV STATE	before_new	before_update	before_destroy
POSYNE STATE	after_create	after_update	after_destroy
	after_save	after_new	

← INTERAKCIJA S BAZOM

```

class AdminUser < ActiveRecord::Base
  before_save :create_hashed_password
  private
    attr_accessor :password
    before_save :create_hashed_password
    after_save :clear_password
  end
end

model validacija u modelu
  • attr_accessor :password → non database attribute → ne sprena se u
    bazu, ali se
    koristi
    user.password
  • before_save :create_hashed_password
  • after_save :clear_password
  } callbacks
    ↳ id je potreban za validaciju
    before i za NEW i za UPDATE

  • - private
    → SVE ISPOD OVOGA JE METODE NOGU ZVAT SAMO IZ PLE KLASICE
    - def create_hashed_password
      ↳ METODA STVARA HASHEO PASSWORDA
      unless password.blank?
        self.salt = AdminUser.make_salt(username) if salt.blank?
        self.hashed_password = AdminUser.hash_with_salt(password, salt)
      end
    end

    - def clear_password
      self.password = nil
    end

  • validates_length_of :password, :within => 8..25, :on => :create

```

AUTENTIFIKACIJA

↳ u modelu

- def self.authenticate(username = "", password = "")
 user = AdminUser.find_by_username(username)
 if user & user.password_match?(password)
 return user
 else
 return false
 end
- def password_match?(password = "")
 hashed_password == AdminUser.hash_with_salt(password, salt)

METODA KOJA VRACA BOOLEANOVU VRIJEDNOST (TRUE, FALSE)

LOGIN & LOGOUT

↳ n kontrolev

generis
ordmeh: template

- nits generete controller access menu login
- dodat akcije def index , attempt_login , logout
 - menu render('menu')

[dodat na nih layout 'admin'
- menu.html.erb → wredit da ima linkove koja trebaju imati
- login → form-tag forma koja ima action attempt_login
- routet config/routes.rb


```
match 'admin':to => 'access#menu'
```
- def attempt_login

```
authorized_user = AdminUser.authenticate(params[:username], params[:password])
if authorized_user
```

```
  flash[:notice] = "You are logged in"
```

```
  redirect_to(:action => 'menu')
```

[session[:user_id] =
 authorized_user.id
session[:username] =
 authorized_user.username

```
else
  flash[:notice] = "Neispravno korisnicko ime ili lozinka"
  redirect_to(:action => 'login')
end
```

```
• def logout
```

```
  flash[:notice] = "You have logged out"
```

[session[:user_id] = nil
session[:username] = nil

```
end
```

COOKIES & SESSIONS

↳ hukijni ~ 4 kB

cookies [:username] = "jsmith"

cookies [:username] = { :value => "jsmith", :expires => 1 week from now }

SESSIONS

session [:username] = "jsmith"

↳ session-cookies => 4 kB, enkriptirano

↳ cookie storage THE BEST

/config / initializers + session-store.rb
+ secret_token.rb

→ određujemo ime cookie i gdje se
 menjaju session varijable

→ mafi key za enkripciju cookiea

- primjer koristjenja u controlleru:

session[:user_id] = authorized_user.id

RESTRICTING ACCESS - BEFORE FILTERS

- callbacks im na modelu - ekskluzivno

- before filters im na kontroleru ekskluzivno

• class AccessController < ApplicationController

before_filter :confirm_logged_in, → IDE, v

:except => [:login, :attempt_login, :logout]

:only => [:method]

private → TREBAJU BIT PRIVATE
def confirm_logged_in DA SE NE MOGU POZIVAT KAO AKCIJE

ISPOD LAYOUT 'ADMIN'
KONTROLERU

• must return false to halt filter chain:

- private

- def confirm_logged_in

unless session[:user_id]

flash[:notice] = "Please log in."

redirect_to(:action => 'login')

return false

else

return true

end

- confirm-logged-in metode + private ide u Application Controller
 (da mi kontroler mogu koristit)
 → postje: protected
- dodat:
 before-filter : confirm-logged-in
 za sve kontrolere
 ↳ primjerit:
redirect_to(:controller => 'access', :action => 'login')

OVO ODRŽAVAJU JER
 AKCIJA LOGIN POSTAVI SMO U DRUG KONTROLERU

14 IMPROVING THE SIMPLE CMS

NESTING

↳ mi redirekti moraju račinat u "višeg" objektu

SUBJECTS / LIST.HTML.ERB

L%> link_to("View Pages", {:controller => 'pages', :action => 'list', :id => subject.id},
 class => 'action show') %>

PAGES CONTROLLER
 • before-filter : find-subject

```
private
def find-subject
  if params[:subject_id]
    @subject = Subject.find-by-id(params[:subject_id])
  end
end
```

- def list dodat
... .where(:subject_id => @subject.id)

- def new imenit
@page = Page.new(:subject_id => @subject.id)

- def create imenit
redirect_to(:action => 'list', :subject_id => (@page.subject_id))

OVO ODRŽAVAJU
POSTAVI

AKO JE
SUBJECT IZNJEDRIV

```
def update :isogenit
  redirect_to(:action => 'show', :id => @page.id, :subject_id => @page.subject_id)
```

```
def destroy :isogenit
  redirect_to(:action => 'list', :subject_id => @subject.id)
```

- u vim vremma dodat u linkove .., :subject_id => @subject.id }
- u pages/list.html.erb dodat link na > subject
↳ = link_to("« Back to Subjects List", {:controller => "subjects",
:action => 'list'}, :class => 'back-link') %>

MODULI

lib / position-mover.rb → voleme radi!

```
module PositionMover
  def move_to_position(new_position)
    end
end
```

app / models / subject.rb

```
class Subject < ActiveRecord::Base
  include PositionMover
```

← MODULE JE KAO
INSTANCIJAT U MODELIMA
(ISTO KAO DA SNO KOPIRALI KOĐ)

- rails loads me iz app/helpers, app/models/, lib/ → NERATO,

• ZA INCUDIT MODUL

```
↳ spring class u modelu
  require 'lib/position-mover'
```

↳ nekon class

```
include PositionMover
```

PUBLIC AREA

- ↳ controller Public index show
- ↳ novi layout public.html.erb
- ↳ index.html.erb je staticen tekot
- ↳ def show controller da prikazuje stvari s :idem
- ↳ routes => match 'show/:id', :to => 'public#show'
 - ↳ OVDJE JE STRING
 - ↳ BILO ČTO OVOJE POSTAJE ID
 - ↳ PROVJERA OVDJE
- ↳ napravit show rich
- ↳ promjenjiti glavni route u aplikaciji

PUBLIC AREA NAVIGATION

subjects.all = [] → pretvara objekte u array → tako da ne baca grešku ako je prezna (nil)

link_to unless current → isto kao link_to

ERRORS

LOG FILES

- logger.debug ("The name is #{@subject.name}")
- logger.info ("Starting the subject update ...")
- logger.warn ("Invalid log in attempted for #{params[:username]}")
- logger.error ("Page with permalink #{params[:id]} not found")
- logger.fatal ("Position Mover module not loaded")

DEBUGGING

- `object.inspect`
- `Li> = debug(object) ;>`
- `puts "/* Got here */"`
- `rdebug -d`

GREŠKE NA PRODUKCIJI

config/environments/production.rb

`config.consider_all_requests_local = false` → pokazuje error pages

- exception notification

↳ sajte email kjer se dogodi greška na aplikaciji

public/404.html

/500.html

/422.html

↳ GIT ili GEM

EXCEPTION-NOTIFICATION

DEPLOYING

1. ISP na Rails

→ HEROKU, JOVENT, LINODE, RAILS MACHINE, ENGINEYARD
RAILS PLAYGROUND, RIMUHOSTING, SLICE HOST

2. INSTALL software na server

→ RUBY, RUBY GEMS, RAILS, MYSQL, MYSQL GEM

↳ SSH `username@server.com`

3. choose web server

FULL-FEATURED:

APACHE
NGINX
LIGHTTPD

LAGOM:

PASSENGER / MOD-RAILS
MONGREL
THIN
UNICORN

4. get deployment tools



FTP, GIT

DEPLOY SCRIPTING

↳ CAPISTRANO [HTTP://CAPIFY.COM](http://CAPIFY.COM)

APACHE + PASSENGER