



**Universidad Nacional de Río Cuarto**  
**Facultad de Cs. Exactas Físico Químicas y Naturales**  
**Departamento de Computación**  
**Ingeniería de Software (Cód. 3304) - 2024**

## **Taller IS 2024 - Actividades**

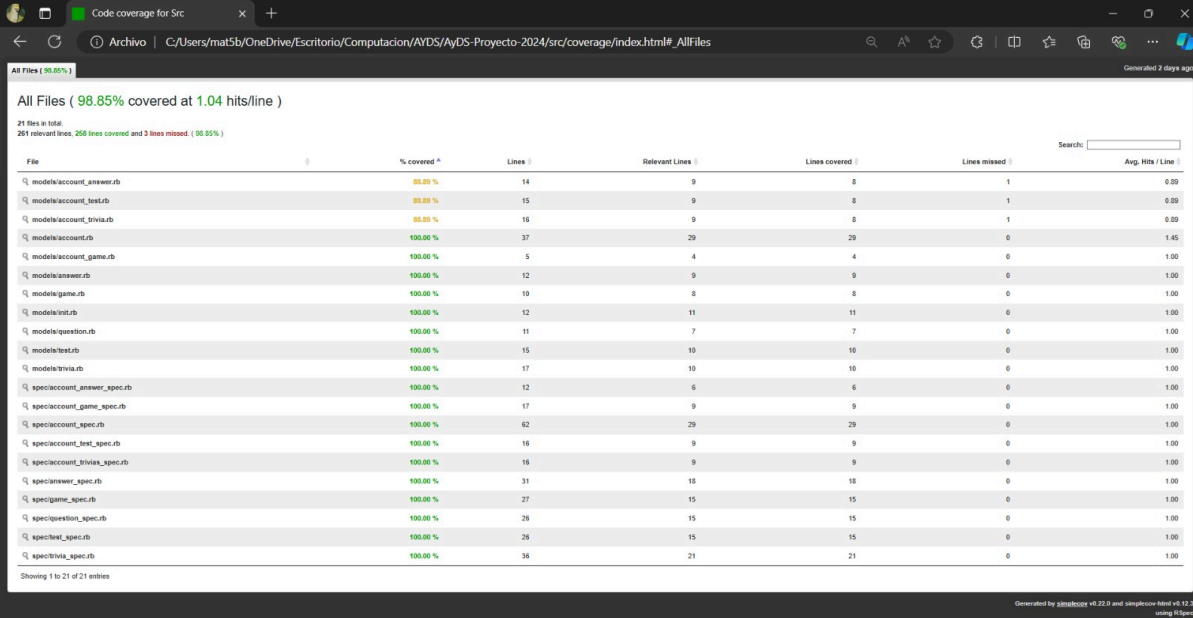
Integrantes:

- Bavera Guillermo
- Bricco Matias
- Irigoyen Juan Cruz

# Actividad Nro 1 : Cobertura

Para esta primera actividad, realizamos un análisis de cobertura de pruebas en nuestra plataforma web, **KnowAboutEsports**. El informe de cobertura generado ofrece una visión detallada del nivel actual de pruebas en nuestro código.

A continuación, se presentan los resultados obtenidos y las áreas clave que requieren atención para optimizar la calidad y robustez del software



File	% covered	Lines	Relevant Lines	Lines covered	Lines missed	Avg. Hits / Line
model/account_answer.rb	88.89 %	14	9	8	1	0.57
model/account_test.rb	88.89 %	15	9	8	1	0.53
model/account_trivia.rb	88.89 %	16	9	8	1	0.59
model/account.rb	100.00 %	37	29	29	0	1.45
model/account_game.rb	100.00 %	5	4	4	0	1.00
model/answer.rb	100.00 %	12	9	9	0	1.00
model/game.rb	100.00 %	10	8	8	0	1.00
model/init.rb	100.00 %	12	11	11	0	1.00
model/question.rb	100.00 %	11	7	7	0	1.00
model/test.rb	100.00 %	15	10	10	0	1.00
model/trivia.rb	100.00 %	17	10	10	0	1.00
spec/account_answer_spec.rb	100.00 %	12	6	6	0	1.00
spec/account_game_spec.rb	100.00 %	17	9	9	0	1.00
spec/account_spec.rb	100.00 %	62	29	29	0	1.00
spec/account_test_spec.rb	100.00 %	16	9	9	0	1.00
spec/account_trivia_spec.rb	100.00 %	16	9	9	0	1.00
spec/answer_spec.rb	100.00 %	31	18	18	0	1.00
spec/game_spec.rb	100.00 %	27	15	15	0	1.00
spec/question_spec.rb	100.00 %	26	15	15	0	1.00
spec/test_spec.rb	100.00 %	26	15	15	0	1.00
spec/trivia_spec.rb	100.00 %	36	21	21	0	1.00

Para alcanzar el 100% de cobertura, nos enfocamos en mejorar el porcentaje específicamente en tres clases clave: **account\_answer.rb**, **account\_test.rb**, y **account\_trivia.rb**. Al analizar estas clases, identificamos un patrón común en el código:

```
after_commit :update_progress_account
private

def update_progress_account
  account.update_progress
end
```

Tras identificar esta similitud, decidimos comentar las líneas correspondientes, ya que pertenecen a una funcionalidad aún incompleta que planeamos desarrollar en los próximos sprints. Esta estrategia nos permitió mejorar temporalmente la cobertura al 100%, mientras avanzamos en el desarrollo del proyecto.

All Files ( 100.0% ) Generated about a minute ago

All Files ( 100.0% covered at 1.05 hits/line )

21 files in total.  
240 relevant lines, 240 lines covered and 0 lines missed ( 100.0% )

File	% covered	Lines	Relevant Lines	Lines covered	Lines missed	Avg. Hits / Line
models/account.rb	100.00 %	27	20	20	0	1.45
models/account_answer.rb	100.00 %	15	5	5	0	1.00
models/account_game.rb	100.00 %	5	4	4	0	1.00
models/account_test.rb	100.00 %	15	5	5	0	1.00
models/account_trivia.rb	100.00 %	15	5	5	0	1.00
models/answer.rb	100.00 %	12	9	9	0	1.00
models/game.rb	100.00 %	10	8	8	0	1.00
models/test.rb	100.00 %	12	11	11	0	1.00
models/question.rb	100.00 %	11	7	7	0	1.00
models/test.rb	100.00 %	15	10	10	0	1.00
models/trivia.rb	100.00 %	17	10	10	0	1.00
spec/account_answer_spec.rb	100.00 %	12	5	5	0	1.00
spec/account_game_spec.rb	100.00 %	17	9	9	0	1.00
spec/account_test_spec.rb	100.00 %	52	29	29	0	1.00
spec/account_trivia_spec.rb	100.00 %	15	9	9	0	1.00
spec/answer_spec.rb	100.00 %	15	9	9	0	1.00
spec/answer_spec.rb	100.00 %	21	15	15	0	1.00
spec/game_spec.rb	100.00 %	27	15	15	0	1.00
spec/question_spec.rb	100.00 %	25	15	15	0	1.00
spec/test_spec.rb	100.00 %	25	15	15	0	1.00
spec/trivia_spec.rb	100.00 %	35	21	21	0	1.00

Showing 1 to 21 of 21 entries

Generated by **SimpleCov** v0.22.0 and **SimpleCov** v0.22.0

Además, para los siguientes sprints decidimos implementar las siguientes funcionalidades a nuestra aplicación web:

- **Actualización del progreso del usuario en los juegos:** Lo que planteamos es crear un update en la barra del progreso que tenga el usuario en un juego (La misma se podrá ver en forma gráfica) y si se decide volver a empezar alguna trivia, el progreso se verá afectado, ya que ha vuelto a empezar una de las lecciones.
- **Mejoras en el registro e inicio de sesión:** Específicamente, queremos brindarle al usuario nuevas formas para registrarse a la aplicación, por lo tanto consideramos en implementar que el mismo pueda hacerlo mediante su cuenta de Google.
- **Nuevo juego para aprender:** Teniendo en cuenta que nuestra temática podía expandirse, decidimos que añadiremos un nuevo juego para que el usuario pueda aprender y poner a prueba sus conocimientos con un examen final.
- **Mejoras visuales en la aplicación:** Decidimos mejorar ciertos aspectos visuales de la página para que la misma sea más atractiva y dinámica para el usuario.

## Actividad Nro 4 : Refactorización.

Para esta actividad, realizaremos un informe sobre las diferentes ofensas que detectamos por la herramienta de trabajo “Rubocop” y cómo respondimos a algunos de ellos teniendo en cuenta la técnica de refactorización usada y que tipo de “Code Smell” detectamos. Cabe mencionar que se corrieron los tests del sistema para verificar que los mismos seguían funcionando ante cualquier cambio realizado.

Comenzando con la refactorización realizamos un análisis de ofensas, con la herramienta rubocop, con los siguientes resultados:

```
39 files inspected, 1602 offenses detected, 1555 offenses autocorrectable

Tip: Based on detected gems, the following RuboCop extension libraries might be helpful:
* rubocop-rake (https://rubygems.org/gems/rubocop-rake)
* rubocop-rspec (https://rubygems.org/gems/rubocop-rspec)

You can opt out of this message by adding the following to your config (see https://docs.rubocop.org/rubocop/extensions.html#extension-suggestions for more options):
  AllCops:
    SuggestExtensions: false

7:32 | in AyDS-Proyecto-2024/src
0 x |
0 0 0 3 Sorbet: Disabled Lin. 24, col. 1 Espacios: 2 UTF-8 LF {} Ruby
```

Utilizando la corrección automática logramos reducir esto a:

```
describe Trivia do ...
^^^^^^^^^^^^^^^^^^^^

39 files inspected, 120 offenses detected

Tip: Based on detected gems, the following RuboCop extension libraries might be helpful:
* rubocop-rake (https://rubygems.org/gems/rubocop-rake)
* rubocop-rspec (https://rubygems.org/gems/rubocop-rspec)

You can opt out of this message by adding the following to your config (see https://docs.rubocop.org/rubocop/extensions.html#extension-suggestions for more options):
  AllCops:
    SuggestExtensions: false

7:38 | in AyDS-Proyecto-2024/src
0 x |
0 0 0 3 Sorbet: Disabled Lin. 503, col. 10 Espacios: 2 UTF-8 LF {} Ruby
```

Ahora tocaba refactorizar el código manualmente, comenzamos por las ofensas de los archivos de “models” y “migrate” las cuales detectamos el Code Smell de “**Dispensables**”, concretamente **Comments**.

Código antes:

```
class AccountAnswer < ActiveRecord::Base
  self.table_name = 'account_answers'
  belongs_to :account
  belongs_to :question
  belongs_to :answer
  belongs_to :game

  after_commit :update_progress_account

  def update_progress_account
    account.update_progress
  end
end
```

Añadiendo documentación a cada uno de nuestros modelos y migraciones solucionamos el problema.

```
# frozen_string_literal: true
# The AccountAnswer class represents the relationship between
# an account and an answer. This model tracks the answers selected
# or completed by a user in the system.

class AccountAnswer < ActiveRecord::Base
  self.table_name = 'account_answers'
  belongs_to :account
  belongs_to :question
  belongs_to :answer
  belongs_to :game

  after_commit :update_progress_account

  def update_progress_account
    account.update_progress
  end
end
```

Luego en nuestro archivo seeds.rb encontramos dos tipos de ofensas con categoría **Cambio Divergente** de Code Smell. Uno de ellas era de **Naming Conventions** con respecto al nombre de algunas variables, ya que las mismas no estaban escritas en camelcase. Por lo que, cambiamos el nombre de cada variable como forma de responder antes estas ofensas, brindando un estilo más consistente y legible en el código. Por ejemplo, la variable `game_1` fue renombrada a `game1`.

El otro tipo de ofensa que encontramos se concentraba en nuestras definiciones de trivias para cada juego. Las mismas contenían líneas de longitud grande (**Large Lines**) lo cual afectaba a la lectura y legibilidad del código. Como técnica de refactorización, realizamos **Extract String** y **Line Break**, es decir dividimos la línea en múltiples partes y luego las concatenamos (En Ruby con el operador “\”). Solucionando las ofensas y dejando el código más legible.

```

Trivia.create(
  number: 2,
  title: 'Neutro',
  description:
    'El Neutro o Juego Neutro se refiere a la situación de la partida en la que ningún jugador tiene ' \
    'iniciativa sobre el otro. Durante el mismo, los jugadores usaran Footsies (Generalmente usando ' \
    'ataques de Poke o buscando el Whiff Punish), Antiaéreos y Zoneo con proyectiles para tener la ' \
    'situación controlada hasta que se rompa esta situación donde alguno de los jugadores obtendrá ventaja ' \
    'sobre el otro, ya sea por haber hecho un Whiff Punish o por simple mala defensa. Estos momentos en ' \
    'partida se presentan en todo momento y sirven para conocer las posibles jugadas que puede tomar el ' \
    'rival, pero también como responder a las mismas, como si fuese un ajedrez mental. ' \
    'En la saga Street Fighter ha estado muy presente ya que en su mayoría de juegos se manifiestan estas ' \
    'situaciones a nivel profesional constantemente y que en su mayoría han dado un análisis profundo de ' \
    'como jugar a nivel alto.',
  test_letter: test_sf.letter,
  mode: 'beginner',
  game_number: game2.number
)

```

También encontramos en el archivo `account_spec.rb` la ofensa **Large Block Length**, ya que el bloque de código superaba el límite de líneas mínimas.

Antes de refactorizar tenía la siguiente forma:

```

# frozen_string_literal: true

ENV['APP_ENV'] = 'test'
require File.expand_path('../models/init', __dir__)
require 'rspec'
require 'rack/test'
require 'spec_helper'

describe Account do
  describe 'validations' do
    let(:valid_attributes) do
      {
        name: 'Juan',
        email: 'juandoe@gmail.com',
        password: 'password1',
        nickname: 'juanito123',
        progress: 0
      }
    end

    context 'with valid attributes' do
      it 'is valid' do
        account = Account.new(valid_attributes)
        expect(account).to be_valid
      end
    end

    context 'with invalid attributes' do
      it 'is invalid due to incorrect email format' do
        invalid_email_account = Account.new(valid_attributes.merge(email:

```

```

'invalid_email.com'))
  expect(invalid_email_account.valid?).to be_falsey
end

it 'is invalid due to short password' do
  short_password_account = Account.new(valid_attributes.merge(password:
'12'))
  expect(short_password_account.valid?).to be_falsey
end

it 'is invalid due to invalid nickname' do
  invalid_nickname_account = Account.new(valid_attributes.merge(nickname:
'ju anito'))
  expect(invalid_nickname_account.valid?).to be_falsey
end

it 'is invalid due to invalid name' do
  invalid_name_account = Account.new(valid_attributes.merge(name:
'Juan1'))
  expect(invalid_name_account.valid?).to be_falsey
end
end
end

describe 'uniqueness' do
  it 'is invalid if an account with the same email already exists' do
    Account.create!(name: 'Juan', email: 'juanito@gmail.com', password:
'Juanito32', nickname: 'juanito')
    duplicate_account = Account.new(name: 'Juan', email: 'juanito@gmail.com',
password: 'Juanito32', nickname: 'juanito2')
    expect(duplicate_account.valid?).to be_falsey
  end
end
end

```

Esta ofensa la solucionamos utilizando la técnica **Extract Method**, la cual consiste en tomar un fragmento de código y convertirlo en un método independiente con un nombre que explique su propósito. Esto ayuda a reducir la longitud de los métodos y mejora la legibilidad del código.

Por lo tanto, lo que hicimos fue dividir el archivo `account_spec.rb` en dos:

- El primero sería `account_validations_spec.rb`, el cual contiene los distintos tipos de test de validaciones para el registro, como por ejemplo que el email sea correcto, entre otros.
- El segundo sería `account_uniqueness_spec.rb`, el cual contiene el método de testeo para ver si una cuenta era única.

Quedando de la siguiente manera:

- account\_validations\_spec.rb

```
# frozen_string_literal: true

ENV['APP_ENV'] = 'test'
require File.expand_path('../models/init', __dir__)
require 'rspec'
require 'rack/test'
require 'spec_helper'

describe Account, 'validations' do
  let(:valid_attributes) do
    { name: 'Juan', email: 'juandoe@gmail.com', password: 'password1', nickname:
'juanito123', progress: 0 }
  end

  describe 'with valid attributes' do
    subject { Account.new(valid_attributes) }

    it 'is valid' do
      expect(subject).to be_valid
    end
  end

  describe 'with invalid attributes' do
    describe 'when email format is invalid' do
      subject { Account.new(valid_attributes.merge(email: 'juanitoyieil.com')) }

      it 'is invalid' do
        expect(subject.valid? && subject.correct_format_of_fields?).to be_falsey
      end
    end

    describe 'when password is too short' do
      subject { Account.new(valid_attributes.merge(password: 'ju')) }

      it 'is invalid' do
        expect(subject.valid? && subject.correct_format_of_fields?).to be_falsey
      end
    end

    describe 'when nickname contains spaces' do
      subject { Account.new(valid_attributes.merge(nickname: 'ju anito')) }

      it 'is invalid' do
        expect(subject.valid? && subject.correct_format_of_fields?).to be_falsey
      end
    end
  end
end
```



```

describe 'when name contains numbers' do
  subject { Account.new(valid_attributes.merge(name: 'Juanit0')) }

  it 'is invalid' do
    expect(subject.valid? && subject.correct_format_of_fields?).to be_falsey
  end
end
end
end

```

- account\_uniqueness\_spec.rb

```

# frozen_string_literal: true

ENV['APP_ENV'] = 'test'
require File.expand_path('../models/init', __dir__)
require 'rspec'
require 'rack/test'
require 'spec_helper'

describe Account, 'uniqueness validation' do
  let(:valid_attributes) do
    { name: 'Juan', email: 'juandoe@gmail.com', password: 'password1', nickname: 'juanito123', progress: 0 }
  end

  it 'is invalid due to duplicate email' do
    Account.create(valid_attributes)
    duplicate_account = Account.new(valid_attributes)

    expect(duplicate_account.save).to be_falsey
  end
end

```

Por último, en nuestro archivo server.rb se concentraban la gran mayoría de ofensas y al analizar el código detenidamente, encontramos muchos Code Smell los cuales debían ser solucionados mediante diferentes técnicas de refactorización. Estos fueron:

- **Shotgun Surgery**
- **Cambio Divergente (Long Method, Naming Conventions, etc.)**
- **Bloaters**
- **Dispensables (Comments, Dead Code)**

Para resolver estas ofensas utilizamos **Extract Variable** (Porque teníamos varias expresiones donde no era claro lo que se calculaba en una línea y decidimos hacerlas por partes para que sea más comprensible y descriptivo a la hora de entender su propósito.), **Split Temporary Variable** (Para prevenir problemas a futuro, aplicamos esta técnica a variables que eran utilizadas de manera repetida en un método para luego generar un Extract Method de manera sencilla, además de

que el código quedaría más descriptivo y con mayor facilidad para su mantenimiento.), **Extract Method** (Muchos de nuestros métodos llegaban a ser largos, entonces tomar partes de los mismos y colocarlas en métodos por separado, cada uno de ellos con un nombre que los describa correctamente. Esto también nos ayudó para que el código sea más claro, mantenible y reutilizable a futuro.), **Reorder Method** (Esta técnica fue usada para mover los métodos de una clase para reorganizar el código y que quedé con una mejor comprensión y legibilidad para su lectura.) y **Temp with Query** (Al igual que con Extract Method, este fue utilizado para disminuir la longitud de métodos.).

Como conclusión, logramos reducir las ofensas a 13, quedando simplemente algunas que pasan el límite de renglones o parámetros, solamente por uno o dos valores y mantener un índice alto de cobertura de nuestros test con estos cambios realizados.