

# Static Testing Tools and White-box test design

---

## Static Testing

---

### Rubocop

Linting and style enforcement. Rules are defined in `./server/.rubocop.yml`.

```
# analyze project
bundle exec rubocop

# Run analyzer and make rubocop automatically fix linting issues
bundle exec rubocop -a
```

#### Successful rubocop:

```
bundle exec rubocop
Inspecting 114 files
.....
.....
114 files inspected, no offenses detected
.rubocop.yml: Style/SpaceInsideArrayLiteralBrackets has the wrong namespace -
replace it with Layout/SpaceInsideArrayLiteralBrackets
```

#### Linting errors highlighted by rubocop

```
bundle exec rubocop
Inspecting 114 files
.....C.....
Offenses:

app/controllers/api/v1/auth_controller.rb:10:19: C: [Correctable]
Layout/SpaceInsideReferenceBrackets: Do not use space inside reference
brackets.

  code = params[ :code ]
          ^
app/controllers/api/v1/auth_controller.rb:10:25: C:
Layout/SpaceInsideReferenceBrackets: Do not use space inside reference
brackets.

  code = params[ :code ]
          ^
app/controllers/api/v1/auth_controller.rb:18:19: C: [Correctable]
Style/StringLiterals: Prefer single-quoted strings when you don't need string
interpolation or special symbols.

  redirect_to "https://munchora.pro/home"
          ^^^^^^^^^^^^^^^^^^^^^^^^^^

114 files inspected, 3 offenses detected, 2 offenses autocorrectable
.rubocop.yml: Style/SpaceInsideArrayLiteralBrackets has the wrong namespace -
replace it with Layout/SpaceInsideArrayLiteralBrackets
```

# SonarQube

SonarQube (<https://www.sonarsource.com/>) is a *static code analysis tool* that automatically inspects code for bugs, vulnerabilities, code smells, and test coverage without running the program.

SonarQube can be run through the use of Docker (<https://medium.com/@index23/start-sonarqube-server-and-run-analyses-locally-with-docker-4550eb7112a3>)

```
cd ./server  
docker-compose -f docker-compose-sonar-qube.yml up
```

Go to **SonarQube dashboard** on <http://localhost:9000> - default credentials are login: admin password: admin

Use following command to scan project with SonnarScanner:

```
docker run \
--rm \
-v "$(pwd):/usr/src" \
--network="host" \
-e SONAR_HOST_URL="http://localhost:9000" \
-e SONAR_SCANNER_OPTS="--sonar.projectKey=server --sonar.sources=./ --sonar.test=test --sonar.javascript.lcov.reportPaths=test/coverage/lcov.info" \
-e SONAR_TOKEN="${SONAR_TOKEN}" \
sonarsource/sonar-scanner-cli
```

Sonarqube do also provide a cloud based solution which can be included in CI pipeline:

The screenshot shows the SonarQube Main Branch Summary for the project 'exam-testing-munchora'. The summary indicates a 'Passed' status. Key metrics displayed include:

- Security: 0 issues (A grade)
- Reliability: 8 issues (C grade)
- Maintainability: 19 issues (A grade)
- Hotspots Reviewed: 100% (1.5k Lines of Code)
- Coverage: 50.6%
- Duplications: 0.0%

The 'Overall Code' tab is selected. The 'Passed' status is highlighted with a green checkmark icon. Other tabs available are 'New Code' and 'Overall Code'.

# White Box Design Techniques

Focuses on the code and the structural elements.

**Statement Coverage:** Measures whether each line of code has been executed by the test suite at least once. **decision coverage:** Measures whether **each decision (true/false outcome) of every conditional statement** has been exercised at least once.

The code coverage is collected by `rspec` and `simplecov`, so whenever test command `bundle exec rspec` is executed a coverage report is generated in JSON and HTML located at `./server/coverage/index.html` showing line-by-line coverage and totals.

Default `simplecov` measures coverage by line coverage which can be an issue with ternary operator/one line conditionals `number.odd? ? "odd" : "even"`

it can be set to use branch coverage instead:

```
SimpleCov.start do
  enable_coverage :branch
end
```

All Files ( 19.32% )	Controllers ( 18.73% )	Channels ( 0.0% )	Models ( 57.41% )	Mailers ( 0.0% )	Helpers ( 100.0% )	Jobs ( 0.0% )	Libraries ( 100.0% )	Ungrouped ( 7.96% )	Generated less than a minute ago
All Files ( 19.32% ) covered at 25.35 hits/line )									
55 lines in total. 1258 relevant lines, 243 lines covered and 1015 lines missed. ( 19.32% )									
Search: <input type="text"/>									
File	% covered ^	Lines	Relevant Lines	Lines covered	Lines missed	Avg. Hits / Line			
<code>app/channels/application_cable/channel.rb</code>	0.00 %	4	4	0	4	0.00			
<code>app/channels/application_cable/connection.rb</code>	0.00 %	61	47	0	47	0.00			
<code>app/channels/notifications_channel.rb</code>	0.00 %	9	7	0	7	0.00			
<code>app/controllers/api/v1/feedbacks_controller.rb</code>	0.00 %	52	40	0	40	0.00			
<code>app/controllers/api/v1/grocery_list_audits_controller.rb</code>	0.00 %	8	7	0	7	0.00			
<code>app/controllers/api/v1/grocery_list_item_audits_controller.rb</code>	0.00 %	8	7	0	7	0.00			
<code>app/controllers/api/v1/grocery_lists_controller.rb</code>	0.00 %	112	91	0	91	0.00			
<code>app/controllers/api/v1/invoices_controller.rb</code>	0.00 %	31	22	0	22	0.00			
<code>app/controllers/api/v1/item_usages_controller.rb</code>	0.00 %	2	2	0	2	0.00			
<code>app/controllers/api/v1/recipe_suggestions_controller.rb</code>	0.00 %	15	14	0	14	0.00			
<code>app/controllers/api/v1/recipe_summary_controller.rb</code>	0.00 %	8	7	0	7	0.00			
<code>app/controllers/api/v1/recipes_controller.rb</code>	0.00 %	238	197	0	197	0.00			
<code>app/controllers/api/v1/subscriptions_controller.rb</code>	0.00 %	46	34	0	34	0.00			
<code>app/controllers/api/v1/text_controller.rb</code>	0.00 %	11	10	0	10	0.00			
<code>app/controllers/api/v1/user_audits_controller.rb</code>	0.00 %	8	7	0	7	0.00			
<code>app/errors/item_usage_limit_exceeded.rb</code>	0.00 %	1	1	0	1	0.00			
<code>app/jobs/application_job.rb</code>	0.00 %	7	2	0	2	0.00			
<code>app/mailers/application_mailer.rb</code>	0.00 %	4	4	0	4	0.00			
<code>app/models/feedback.rb</code>	0.00 %	10	9	0	9	0.00			
<code>app/models/grocery_list_audit.rb</code>	0.00 %	2	2	0	2	0.00			
<code>app/models/grocery_list_item_audit.rb</code>	0.00 %	2	2	0	2	0.00			
<code>app/models/invoice.rb</code>	0.00 %	30	23	0	23	0.00			
<code>app/models/recipe_comment.rb</code>	0.00 %	4	4	0	4	0.00			
<code>app/models/recipe_like.rb</code>	0.00 %	4	4	0	4	0.00			
<code>app/models/recipe_suggestion.rb</code>	0.00 %	4	4	0	4	0.00			
<code>app/models/recipe_summary_view.rb</code>	0.00 %	8	6	0	6	0.00			
<code>app/models/subscription.rb</code>	0.00 %	7	6	0	6	0.00			
<code>app/models/subscription_plan.rb</code>	0.00 %	8	7	0	7	0.00			
<code>app/models/user_audit.rb</code>	0.00 %	2	2	0	2	0.00			
<code>app/services/auth/google_auth_service.rb</code>	0.00 %	60	49	0	49	0.00			
<code>app/services/grocery_lists_creator.rb</code>	0.00 %	5	5	0	5	0.00			

## User creation | User #initialize

### Test cases 100% statement coverage

```
#1. first_name="John", last_name="Doe", email="john@doe.com", provider=nil,  
uid=nil, password="secret123"  
    # TRUE path provider.blank? & TRUE :password is_a?(String)  
  
#2. first_name="John", last_name="Doe", email="john@doe.com",  
provider="google", uid="abc123", password=nil  
    # TRUE branch of both provider.present? and uid.present?
```

### Test cases 100% decision coverage

```
#1. first_name="John", last_name="Doe", email="john@doe.com", provider=nil,  
uid=nil, password="secret123"  
    # TRUE path for provider.blank? and TRUE for :password is_a?(String)  
  
#2. first_name="John", last_name="Doe", email="john@doe.com", provider=nil,  
uid=nil, password=1234  
    # FALSE branch of :pasword is_a?(String)  
  
#3. first_name="John", last_name="Doe", email="john@doe.com",  
provider="google", uid=nil, password=nil  
    # FALSE branch of uid.present? when provider.present?  
  
#3. first_name="John", last_name="Doe", email="john@doe.com",  
provider="google", uid="abc123", password=nil  
    # TRUE branch of both provider.present? and uid.present?
```