

Software Evolution and Maintenance

FCAI-CU-SWE 2022

You are given a Java project University Timetabling (<https://sourceforge.net/projects/unitime/>) as a base version, you are the maintenance team that needs to implement changes to the existing software. We will only focus on the Java backend.

The change request to the project can either be a bug fix, change in an existing feature or addition of a new requirement. Motivate the need for the change that you have proposed.

Project Implementation

- You should work in teams of no more than 4 and no less than 3 From the same lab.
- The project is organized into two phases**

Phase 1:

- Create a GitHub repository and all team members should be contributors. contributors (Note: All of your work should be pushed to GitHub as pull requests and reviewed by another team member.
- Provide a description of what the project does.
- Run static code analysis tool and generate an analysis report (You may use SonarQube as we did in the lab)
- Provide a description for the feature/bug fix that you will start working on it.
- Register the change request you defined in step 4 to a free ticketing system or software such as (Bugzilla, Jira, Agiloft, ...),
- You are requested to update it through the lifetime of the change request to indicate it's status, and any changes that are related.

- Deliverables**

- A link to your Github repo
- A link to your ticketing system
- Static Analysis Report For java backend code
- text file (word or pdf) containing the description of what the project does. Mention what technique(s) you used to obtain this description
- A description for the feature/bug fix that you will start working on it.

- Notes:**

- All of the required deliverables will be pushed to your Github repository, all team members must use Github to push their tasks.
- You should submit a text or pdf file with your names and github repo (don't push anything after deadline),
- the file should have the following format file name:
Group_1stStudentId_2ndStudentId_3rdStudentId_4thStudentId

Team Member:

ID	Name	GitHub Username
20196037	Amr Halaby	AmrHalaby
20196026	Sohaila Gamal	Sohailla
20206023	Dalia Gamal	DaliaGamal11
20196003	Ahmed Gamal	Ahmedgemy2002@gmail.com

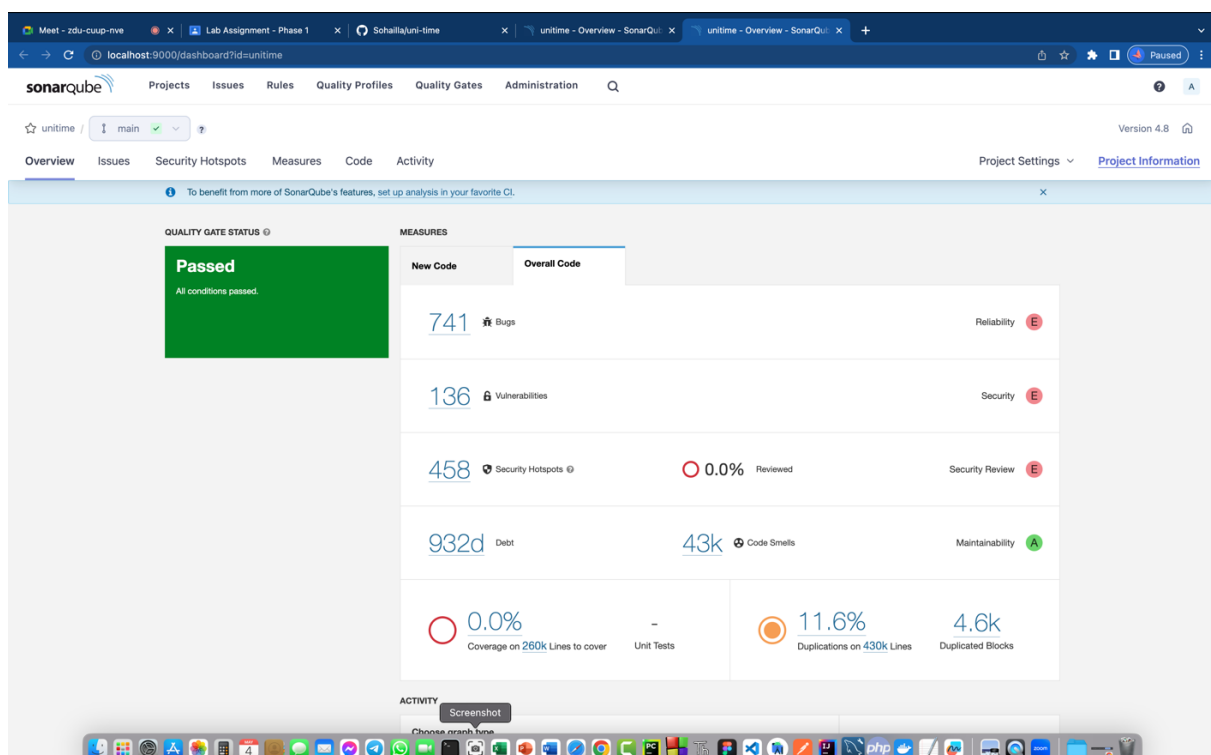
- **GitHub Repository:** <https://github.com/Sohailla/uni-time>
- **Ticketing System [Trello]:** <https://trello.com/b/nGqGdbd8/uni-time>
- **Static Analysis Report For java backend code**

The screenshot shows an IDE window titled 'unitime - build.properties'. The editor displays the 'build.properties' file with the following content:

```
# The Apache Foundation licenses this file to you under the Apache License,
# Version 2.0 (the "License"); you may not use this file except in
# compliance with the License. You may obtain a copy of the License at:
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
```

The terminal window shows the following build logs:

```
[INFO] Sensor Zero Coverage Sensor
[INFO] Sensor Zero Coverage Sensor (done) | time=318ms
[INFO] Sensor Java CPD Block Indexer
[INFO] Sensor Java CPD Block Indexer (done) | time=2008ms
[INFO] SCM Publisher SCM provider for this project is: git
[INFO] SCM Publisher 2381 source files to be analyzed
[INFO]
[INFO] SCM Publisher 2381/2381 source files have been analyzed (done) | time=13602ms
[INFO] CPD Executor 504 files had no CPD blocks
[INFO] CPD Executor Calculating CPD for 1741 files
[WARNING] Too many duplication groups on file JavaSource/org/unitime/timetable/onlinesectioning/custom/purdue/PurdueCourseRequestsValidati
onProvider.java. Keep only the first 100 groups.
[INFO] CPD Executor CPD calculation finished (done) | time=444ms
[INFO] Analysis report generated in 593ms, dir size=54.9 MB
[INFO] Analysis report compressed in 6738ms, zip size=19.3 MB
[INFO] Analysis report uploaded in 338ms
[INFO] ANALYSIS SUCCESSFUL, you can find the results at: http://localhost:9000/dashboard?id=unitime
[INFO] Note that you will be able to access the updated dashboard once the server has processed the submitted analysis report
[INFO] More about the report processing at http://localhost:9000/api/ce/task?id=AYfoP8-j2wEp87Dcw44q
[INFO] Analysis total time: 5:32.902 s
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 11:05 min
[INFO] Finished at: 2023-05-04T21:31:50+02:00
[INFO] -----
+ unitime git:(master) x
```



sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration

unitime / main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

1 / 43,699 issues 952d effort

Filters

Issues in new code

Type

- Bug 741
- Vulnerability 136
- Code Smell 43k

Severity

- Blocker 199
- Critical 12k
- Major 14k
- Minor 17k
- Info 96

Scope

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

Assignee

Author

Remove this commented out code. 9 years ago L88 unused

Remove this commented out code. 11 years ago L75 unused

Remove this commented out code. 5 years ago L60 unused

Remove this commented out code. 9 years ago L128 unused

Complete the task associated to this "TODO" comment. 6 years ago L176 cwe

Complete the task associated to this "TODO" comment. 6 years ago L157 cwe

Remove this commented out code. 3 years ago L107 unused

Remove this commented out code. 13 years ago L363 unused

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration

unitime / main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

1 / 43,699 issues 952d effort

Filters

Issues in new code

Type

- Bug 741
- Vulnerability 136
- Code Smell 43k

Severity

- Blocker 199
- Critical 12k
- Major 14k
- Minor 17k
- Info 96

Scope

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

Assignee

Author

Remove this commented out code. 9 years ago L826 unused

Remove this commented out code. 15 years ago L57 unused

Remove this commented out code. 11 years ago L3662 unused

Remove this commented out code. 11 years ago L3751 unused

Remove this commented out code. 9 years ago L67 unused

Remove this commented out code. 6 years ago L71 unused

Remove this commented out code. 11 years ago L74 unused

Remove this commented out code. 10 years ago L645 unused

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration

unitime / main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

1 / 43,699 issues 952d effort

Filters

Issues in new code

Type

- Bug 741
- Vulnerability 136
- Code Smell 43k

Severity

- Blocker 199
- Critical 12k
- Major 14k
- Minor 17k
- Info 96

Scope

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

Assignee

Author

JavaSource/org/unitime/commons/Calendar/TimeZoneGenerator.java

Change this "try" to a try-with-resources. 4 months ago L71

Define and throw a dedicated exception instead of using a generic one. 5 years ago L77

JavaSource/org/unitime/commons/Debug.java

Add a private constructor to hide the implicit public one. 15 years ago L38

A "NullPointerException" could be thrown; "e" is nullable here. 15 years ago L47

Define a constant instead of duplicating this literal "ERROR:" 4 times. 15 years ago L97

Define a constant instead of duplicating this literal "WARNING:" 4 times. 15 years ago L100

Define a constant instead of duplicating this literal "INFO:" 4 times. 15 years ago L103

Provide the parametrized type for this generic. 15 years ago L131

0 is a valid index, but is ignored by this check. 15 years ago L134

JavaSource/org/unitime/commons/Email.java

Convert the abstract class "Email" into an interface. 10 years ago L33

sonarqube

Projects Issues Rules Quality Profiles Quality Gates Administration

unitime / main

Overview Issues Security Hotspots Measures Code Activity

Project Settings Project Information

1 / 43,699 issues 952d effort

Filters

Issues in new code

Type

- Bug 741
- Vulnerability 136
- Code Smell 43k

Severity

- Blocker 199
- Critical 12k
- Major 14k
- Minor 17k
- Info 96

Scope

Resolution

Status

Security Category

Creation Date

Language

Rule

Tag

Directory

File

Assignee

Author

JavaSource/org/unitime/commons/Email.java

Convert the abstract class "Email" into an interface. 10 years ago L33

Define and throw a dedicated exception instead of using a generic one. 10 years ago L35

Define and throw a dedicated exception instead of using a generic one. 10 years ago L39

Define and throw a dedicated exception instead of using a generic one. 10 years ago L41

Define and throw a dedicated exception instead of using a generic one. 10 years ago L43

Define and throw a dedicated exception instead of using a generic one. 9 years ago L45

Define and throw a dedicated exception instead of using a generic one. 10 years ago L47

Define and throw a dedicated exception instead of using a generic one. 10 years ago L49

Define and throw a dedicated exception instead of using a generic one. 10 years ago L51

Define and throw a dedicated exception instead of using a generic one. 1 month ago L53

Define and throw a dedicated exception instead of using a generic one. 1 month ago L71

- **The description of what the project does. Mention what technique(s) you used to obtain this description**

1. What the project does:

UniTime is a comprehensive educational scheduling system that supports developing course and exam timetables, managing changes to these timetables, sharing rooms with other events, and scheduling students to individual classes. It is a distributed system that allows multiple university and departmental schedule managers to coordinate efforts to build and modify a schedule that meets their diverse organizational needs while allowing for minimization of student course conflicts. It can be used alone to create and maintain a school's schedule of classes and/or exams, or interfaced with an existing student information system.

Features

- course timetabling
- examination timetabling
- event management
- student scheduling

2. Maintenance Technique: “Opportunistic Approach”

Opportunistic Approach is a hybrid of the two

1. Begin with top-down, gain an overview of the functions of the program
2. Then selectively apply bottom-up strategies when nearing “code level”
3. Presence of beacons can indicate opportunity for change of strategy

- **A description for the feature/bug fix that you will start working on it.**

We will fix the Email.java and ToolBox.java classes.

In Email.java:

We will modify the code into two classes to support solid principles

In ToolBox.java:

Add new features to class to support availability