CSC 510 Project 2

Tingxiang Ji, tji2 Hui Sun, hsun26 Zhichao Hou, zhou4

Repo link: https://go.ncsu.edu/ra9m1vq
The poster is attacked in the last page.

Notes	Scores 2	Proof
Workload is spread over the whole team (one team member is often Xtimes more productive than the others...	3	Evidence
A track record that everyone is contributing a lot	3	Evidence
Number of commits	3	Evidence
Number of commits: by different people	3	Evidence
Issues reports: there are many	3	Evidence
Issues are being closed	3	Evidence
Docs: doco generated, format not ugly	3	Evidence
Docs: what: point descriptions of each class/function (in isolation)	3	Evidence
Docs: how: for common use cases X,Y,Z mini-tutorials showing worked examples on how to do X,Y,Z	3	Contribution
Docs: why: docs tell a story, motivate the whole thing, deliver a punch-line that makes you want to rush out and use the thing	3	Contribution
Docs: short video, animated, hosted on your repo. That convinces people why they want to work on your code.	3	1. Evidence1 2. Evidence2
Use of version control tools	3	Evidence
Test cases exist	3	Our is 84%+
Test cases are routinely executed	3	Evidence
Issues are discussed before they are closed	3	Example
Chat channel: exists	3	Evidence
Test cases: a large proportion of the issues related to handling failing cases.	3	Evidence
Evidence that the whole team is using the same tools: everyone can get to all tools and files	3	We all commit
Evidence that the whole team is using the same tools (e.g. config files in the repo, updated by lots of different people)	3	Evidence
Evidence that the whole team is using the same tools (e.g. tutor can ask anyone to share screen, they demonstrate the system running on their computer)	3	Wait for Turtor

Evidence that the members of the team are working across multiple places in the code base	3	Evidence
Short release cycles	3	Evidence
The file .gitignore lists what files should not be saved to the repo.	3	Evidence
The file INSTALL.md lists how to install the code	3	Evidence
The file LICENSE.md lists rules of usage for this repo	3	Evidence
The file CODE-OF-CONDUCT.md lists rules of behavior for this repo	3	Evidence
The file CONTRIBUTING.md lists coding standards and lots of tips on how to extend the system without screwing things up	3	Evidence
The file README.md contains all the following	3	Evidence
2min video of new functionality, showing a significant delta from prior.	3	Evidence
DOI badge: exists.	3	Evidence
Badges	3	Evidence

Question	Answer
Question 1.1: Does your website and documentation provide a clear, high-level overview of your software?	Yes
Question 1.2: Does your website and documentation clearly describe the type of user who should use your software?	Yes
Question 1.3: Do you publish case studies to show how your software has been used by yourself and others?	Yes
Question 2.1: Is the name of your project/software unique?	Yes
Question 2.2: Is your project/software name free from trademark violations?	Yes
Question 3.1: Is your software available as a package that can be deployed without building it?	Yes
Question 3.2: Is your software available for free?	Yes
Question 3.3: Is your source code publicly available to download, either as a downloadable bundle or via access to a source code repository?	Yes
Question 3.4: Is your software hosted in an established, third-party repository likeGitHub (https://github.com), BitBucket (https://bitbucket.org),LaunchPad (https://launchpad.net) or-SourceForge (https://sourceforge.net)?	Yes
Question 4.1: Is your documentation clearly available on your website or within your software?	Yes
Question 4.2: Does your documentation include a "quick start" guide, that provides a short overview of how to use your software with some basic examples of use?	Yes
Question 4.3: If you provide more extensive documentation, does this provide clear, step-by-step instructions on how to deploy and use your software?	Yes
Question 4.4: Do you provide a comprehensive guide to all your software's commands, functions and options?	Yes
Question 4.5: Do you provide troubleshooting information that describes the symptoms and step-by-step solutions for problems and error messages?	Yes
Question 4.6: If your software can be used as a library, package or service by other software, do you provide comprehensive API documentation?	Yes
Question 4.7: Do you store your documentation under revision control with your source code?	Yes

Question 4.8: Do you publish your release history e.g. release data, version numbers, key features of each release etc. on your web site or in your documentation?	Yes
Question 5.1: Does your software describe how a user can get help with using your software?	Yes
Question 5.2: Does your website and documentation describe what support, if any, you provide to users and developers?	Yes
Question 5.3: Does your project have an e-mail address or forum that is solely for supporting users?	Yes
Question 5.4: Are e-mails to your support e-mail address received by more than one person?	Yes
Question 5.5: Does your project have a ticketing system to manage bug reports and feature requests?	Yes
Question 5.6: Is your project's ticketing system publicly visible to your users, so they can view bug reports and feature requests?	Yes
Question 6.1: Is your software's architecture and design modular?	Yes
Question 6.2: Does your software use an accepted coding standard or convention?	Yes
Question 7.1: Does your software allow data to be imported and exported using open data formats? e.g. GIF, SVG, HTML, XML, tar, zip, CSV, JSON, NetCDF, or domain specific ones	No
Question 7.2: Does your software allow communications using open communications protocols? e.g. HTTP, FTP, XMPP, SOAP over HTTP, or domain-specific ones	Yes
Question 8.1: Is your software cross-platform compatible? e.g. does it run under two or more of Windows, Unix/Linux and Mac OS X, or can be used from within two or more of Internet Explorer, Chrome, Firefox and Safari?	Yes
Question 9.1: Does your software adhere to appropriate accessibility conventions or standards?	Yes
Question 9.2: Does your documentation adhere to appropriate accessibility conventions or standards?	Yes
Question 10.1: Is your source code stored in a repository under revision control?	Yes
Question 10.2: Is each source code release a snapshot of the repository?	Yes
Question 10.3: Are releases tagged in the repository?	Yes
Question 10.4: Is there a branch of the repository that is always stable? (i.e. tests always pass, code always builds successfully)	Yes
Question 10.5: Do you back-up your repository?	Yes
Question 11.1: Do you provide publicly-available instructions for building your software from the source code?	Yes
Question 11.2: Can you build, or package, your software using an automated tool? e.g. Make (https://www.gnu.org/software/make/), ANT (http://ant.apache.org/), Maven (https://maven.apache.org/), CMake (https://cmake.org/), Python setuptools (https://pypi.python.org/pypi/setuptools), or R package tools (https://cran.r-project.org/doc/manuals/r-devel/R-exts.html)	No
Question 11.3: Do you provide publicly-available instructions for deploying your software?	Yes
Question 11.4: Does your documentation list all third-party dependencies?	Yes
Question 11.5: Does your documentation list the version number for all third-party dependencies?	Yes
Question 11.6: Does your software list the web address, and licences for all third-party dependencies and say whether the dependencies are mandatory or optional?	Yes
Question 11.7: Can you download dependencies using a dependency management tool or package manager? e.g. Ivy (http://ant.apache.org/ivy/), Maven (https://maven.apache.org/), Python pip (https://pypi.python.org/pypi/pip) or setuptools (https://pypi.python.org/pypi/setuptools), PHP Composer (https://getcomposer.org/), Ruby gems (https://rubygems.org), or R PackRat (https://rstudio.github.io/packrat/)	Yes

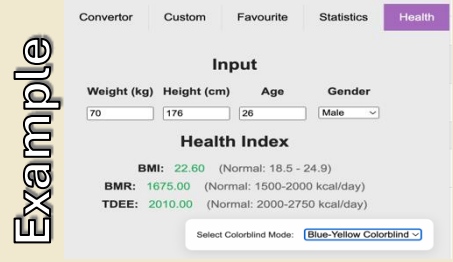
Question 11.8: Do you have tests that can be run after your software has been built or deployed to show whether the build or deployment has been successful?	Yes
Question 12.1: Do you have an automated test suite for your software?	Yes
Question 12.2: Do you have a framework to periodically (e.g. nightly) run your tests on the latest version of the source code?	Yes
Question 12.3: Do you use continuous integration, automatically running tests whenever changes are made to your source code?	Yes
Question 12.4: Are your test results publicly visible?	Yes
Question 12.5: Are all manually-run tests documented?	No
Question 13.1: Does your project have resources (e.g. blog, Twitter, RSS feed, Facebook page, wiki, mailing list) that are regularly updated with information about your software? e.g. release announcements, publications, workshops, conference presentations	Yes
Question 13.2: Does your website state how many projects and users are associated with your project?	No
Question 13.3: Do you provide success stories on your website?	Yes
Question 13.4: Do you list your important partners and collaborators on your website?	Yes
Question 13.5: Do you list your project's publications on your website or link to a resource where these are available?	Yes
Question 13.6: Do you list third-party publications that refer to your software on your website or link to a resource where these are available?	Yes
Question 13.7: Can users subscribe to notifications to changes to your source code repository?	Yes
Question 13.8: If your software is developed as an open source project (and, not just a project developing open source software), do you have a governance model?	Yes
Question 14.1: Do you accept contributions (e.g. bug fixes, enhancements, documentation updates, tutorials) from people who are not part of your project?	Yes
Question 14.2: Do you have a contributions policy?	Yes
Question 14.3: Is your contributions' policy publicly available?	Yes
Question 14.4: Do contributors keep the copyright/IP of their contributions?	Yes
Question 15.1: Does your website and documentation clearly state the copyright owners of your software and documentation?	Yes
Question 15.2: Does each of your source code files include a copyright statement?	Yes
Question 15.3: Does your website and documentation clearly state the licence of your software?	Yes
Question 15.4: Is your software released under an open source licence?	Yes
Question 15.5: Is your software released under an OSI-approved open-source licence?	Yes
Question 15.6: Does each of your source code files include a licence header?	Yes
Question 15.7: Do you have a recommended citation for your software?	Yes
Question 16.1: Does your website or documentation include a project roadmap (a list of project and development milestones for the next 3, 6 and 12 months)?	Yes
Question 16.2: Does your website or documentation describe how your project is funded, and the period over which funding is guaranteed?	No
Question 16.3: Do you make timely announcements of the deprecation of components, APIs, etc.?	No

Smart Converter

Introduction

Smart Converter is an efficient, user-friendly tool designed to handle diverse unit conversions across various categories, statistical analysis and health index report.

Example



Version i

Colorblind Mode: Supports multiple colorblind settings, including red-green, blue-yellow, and total colorblind modes, ensuring usability for visually impaired users.

Statistics Tab: Offers data analysis tools that calculate mean, standard deviation, and entropy, simplifying quick statistical insights.

Health Tab: Computes health indices like BMI, BMR, and TDEE based on weight, height, age, and gender. Normal values are shown in green; out-of-range values appear in red.

Expanded Conversion Library: New unit categories and more options for user-defined conversions.

UI and Performance Improvements: Optimized codebase and refined UI for faster responses and improved mobile compatibility.

Test Case (180+)

Basic Conversion Test: Convert 1 USD to JPY and verify accuracy.

Statistics Test: Input a data set and check mean, standard deviation, and entropy calculations.

Health Index Test: Input user details to calculate BMI, BMR, and TDEE, verifying color-coded output for normal ranges.

Error Handling Test: Input an invalid value to ensure the tool displays an appropriate error message.

....

Version i+1

Multilingual Support: Enable language options to serve a global audience.

Text Enhancement Tools: Add grammar correction and text polishing for streamlined text editing.

Thesaurus Integration: Offer synonyms and related words to aid vocabulary expansion.

Customizable Conversion Settings: Allow users to define conversion parameters for enhanced control.

Voice Input and Output: Implement voice commands for conversions, enhancing hands-free usability.

Demo

Repo

Authors:
Hui Sun
Tingxiang Ji
Zhichao Hou