




COSC2391 Further Programming

S2, 2022

Assignment 2: Project Management Application

	Assessment type: individual assignment; no group work. Submit online via Canvas → Assignments → Assignment 2. Marks are awarded for meeting requirements as closely as possible according to assignment specifications and the supplied rubric. Clarifications/updates may be made via announcements/discussion board.
	Due date: week 13, Friday 29th May at 11:59pm. Late submission penalty will be applied unless special consideration has been granted. There will be three milestones: week 8 in-lab milestone check, week 11 in-lab milestone check and week 14 post submission final interview . You will describe the key concepts adopted in your code, demonstrate code execution, explain your code, and articulate lessons learnt in the milestones. Instructions for milestone checking will be provided separately in the milestone submissions.
	Weighting: 50 marks out of 100 for assignment 2. 3 milestones (4 marks for week 8 in-lab milestone checking, 5 marks for week 11 in-lab milestone checking and 5 marks for week 14 post submission final interview) will add up to 14 marks in addition to the 50 marks.

1. Overview

NOTE: Carefully read this document. In addition, regularly follow the Canvas assignment discussion board for assignment related clarifications and discussion.

In this assignment, you are required to work individually to develop a Project Management Application, named Smart Board, to visually manage your personal projects. You will practice your knowledge of object-oriented design principles, design patterns, Java Collections Framework, generics, exceptions, graphical user interfaces, serialization and deserialization, and unit testing. You will use Java SE 11 or later and JavaFX.

Smart Board is a desktop-based Kanban-style application for managing personal work. It has a similar concept to Trello, which is a web-based Kanban-style application to enable teams to manage any sized collaborative projects from start to finish. You can have a look at Trello website (<https://trello.com/en>) to see how a Trello board looks like.

Task Specification

Basic functional requirements are listed below (mandatory).

- The application can have many users.

- Each user can create a profile, with a unique username, password, the first name and the last name, and the profile photo. If there is no photo, a default profile photo is provided by the application.
- Once the username and password are created, the user can log in.
- Each user has an empty workspace after first login and can perform following actions on the workspace:
 - Edit the profile (i.e., change the first name or the last name, select a new profile photo).
 - Create a new project board representing a personal project to work on. Each project board has a name and columns representing stages in the workflow of the project (e.g., to do, in-progress, completed).
 - Set one project as default and unset the default setting. When multiple projects exist in the workspace, the default project will be the board shown to the user after login. If no project is set as default, the first added project will be the board shown to the user after login.
 - Add a new column to the project board. Each column has a name and a list of tasks.
 - Add a task card to the column. Each task has a name and description.
 - Edit the task name and description.
 - Reorder tasks within a column.
 - Move a task to another column.
 - Rename a project board/column.
 - Delete an existing project board/column/task.
 - Log out (and go back to login page).
- The workspace will display an inspirational quote every time after the user logs in to motivate the user to work on the project. The quote to be displayed will be randomly chosen from a list of quotes at each login and remain the same during the same login period. This list can be hard coded in your application. You are free to choose any quotes that you like (e.g., <https://www.goodreads.com/quotes/tag/motivation>). A minimum of 3 quotes is required to demonstrate the random quote selection at each login.
- The application data should persist between logins for the same user.

Advanced functional requirements are listed below (optional).

- In addition to the name and description, each task can have a due date and a checklist. Each user can perform the following actions:
 - Add a due date by selecting a due date from the calendar.
 - Change or delete the due date or mark it as completed.
 - Add a colour indicator when the due date is approaching or overdue and the task is complete within the due date.
 - Add a checklist and add action items to the checklist.
 - Tick/untick an action item; the completion of the checklist should be reflected in a progress bar.
 - Add a colour indicator when all action items are completed.
 - Edit or delete an action item.

- Delete the checklist.
- The task name, due date, and checklist completion status (number of action items completed) are displayed on the task card.

Please check the Appendix A for a sample interaction with the Smart Board.

Disclaimer: *the specification in this assignment is intended to represent a simplified version of a system in real life and thus is not meant to be a 100% accurate simulation of any real system or service that is being used commercially.*

2. Assessment Criteria

This assessment will determine your ability to implement Object-Oriented Java code according to the specifications shown below. In addition to functional correctness (i.e. getting your code to work) you will also be assessed on code quality. Specifically:

- You are required to demonstrate your understanding of object-oriented programming principles, including encapsulation, abstraction, inheritance, and polymorphism.
- Your object-oriented design should provide high cohesion and low coupling following well established design principles and patterns.
- You are required to use well-tested Java collections.
- You are required to validate all user inputs and catch and handle all exceptions.
- You are required to demonstrate that you have done adequate unit testing using the JUnit framework.
- Your interaction design must include appropriate JavaFX elements to make the system easy-to-navigate and consistent and respond clearly to every user action.
- Your application needs to have a persistent state that must be stored in a file or a database.
- You are required to write properly documented code.
- You should analyse your design choice in building this desktop-based application in a separate one-page report.

3. Learning Outcomes

This assessment is relevant to the following learning outcomes:

CLO1: explain the purpose of OO design and apply the following OO concepts in Java code: inheritance, polymorphism, abstract classes, interfaces and generics.

CLO2: describe and document diagrammatically the OO design of the Java Collection Framework (JCF) and apply this framework in Java code.

CLO3: describe and document diagrammatically the OO design of the JavaFX APIs and apply these APIs to create graphical user interface (GUI) code.

CLO4: demonstrate proficiency using an integrated development environment such as Eclipse for project management, coding and debugging.

CLO5: describe and document diagrammatically common OO design patterns such as Model View Controller (MVC), Singleton, Facade and apply in Java code.

4. Assessment Details

In this assignment, you will incrementally build a project management application, named Smart Board. The assignment is structured into 3 milestones which must be demonstrated to your lab

tutor (adding up to 14 marks in addition to the 50 marks for the assignment). Instructions for milestone checking will be provided separately in the milestone submissions.

Part A (mandatory)

You are required to work on the backend part of the application (excluding data storage, which will be covered in Part C) regarding the basic functions mentioned in the task specification in Section 1 - Overview. You can start this part from week 6 and work on it until the due date.

- You will incorporate object-oriented concepts to design and implement classes (including abstraction, encapsulation, inheritance, and polymorphism).
- You will need to choose appropriate data structures to store relevant information (e.g., a list of tasks in a project; a list of projects in the workspace). You **MUST** use Java Collections Framework.
- You are required to write unit tests for all classes that include functions in the backend part (excluding data storage) using JUnit framework. Your unit tests should have a good coverage of the typical use cases of your classes and test all reasonable corner cases. You do not need to write unit tests for getters and setters.
- You are required to catch and handle all exceptions in your program. Avoid throwing an unspecific Exception; throw a specific exception. You can create custom exception classes if necessary.
- You will need to document the code properly by following Code Conventions for the Java Programming Language by Oracle: <https://www.oracle.com/java/technologies/javase/codeconventions-introduction.html>
- You will adhere to SOLID design principles.

Part B (mandatory)

You are required to work on the frontend part of the application and connect the frontend with the backend regarding the basic functions mentioned in the task specification in Section 1 - Overview. You can start this part from week 7 and work on it until the due date.

- You **MUST** use JavaFX.
- Your program should include appropriate JavaFX components to make the application easy-to-navigate. For example, each window should have a window title; when multiple columns are added to the project board or multiple task cards are added to one column, if they cannot be displayed within the visible area of the board, the user should be able to scroll around the project board to view all columns and tasks within each column.
- Your program should respond clearly to every user action. For example, if the user types a wrong password, the login window should display a clear error message and ask the user to type again; if the user sets a project as default, the default project should be highlighted or changed in a way that notifies the user that it is default now.
- Your program should provide a visual consistency. Avoid using different styles and labels for similar elements on different windows of the application. For example, a button to confirm the completion of renaming is shown as "Ok" in the window for renaming a project and the window for renaming a column.
- You are required to follow MVC design pattern to connect the frontend with the backend.
- You can use other design patterns such as Singleton if necessary.

Part C (mandatory)

You can start this part from week 9 and work on it until the due date.

- Your program should save all application data and allow the application to be restarted in the same state it was in at the end of the previous execution. You can store the data in either a file or a database (using JDBC) when the application closes and restore the data when the application starts again. Note the application may crash during the execution.

- You are required to write a one-page document (minimum 300 words; maximum 500 words) to analyse and justify your design choice in building this desktop-based application. Your analysis and justification will need to cover the following questions:
 - How did you apply MVC design pattern to build this application?
 - How does your code adhere to SOLID design principles?
 - What other design patterns does your code follow? Why did you choose these design patterns?

Part D (optional)

You may optionally implement the advanced functions mentioned in the task specification in Section 1 - Overview. The completion of part D will provide bonus marks (i.e., 5 marks). You can start this part from week 7 if you are planning to obtain some bonus marks.

5. Submission format

- You must submit a zip file of your project via Canvas.
- Include a README with instructions on how to run and test your program.
- You can submit your assignment as many times as you would like prior to the due date. The latest submission will be graded.
- You **MUST NOT** submit compiled files (*.class files). Please check your submission carefully, make sure all java files have been included.

6. Referencing guidelines

You are free to refer to textbooks or notes and discuss the design issues (and associated general solutions) with your fellow students or on Canvas; however, the assignment should be your **OWN WORK**.

The submitted assignment must be your own work. For more information, please visit the academic integrity website: <https://www.rmit.edu.au/students/student-essentials/assessment-and-results/academic-integrity>.

Plagiarism is treated very seriously at RMIT. Plagiarism includes copying code directly from other students, internet or other resources without proper reference. Sometimes, students' study and work on assignments together and submit similar files which may be regarded as plagiarism. Please note that you should always create your own assignment work even if you have very similar ideas.

Plagiarism-detection tools will be used for all submissions. Penalties may be applied in cases of plagiarism.

7. Academic integrity and plagiarism (standard warning)

Academic integrity is about honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarized, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods,
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites.

If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own. RMIT University treats plagiarism as a very serious

offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviors, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

8. Appendix A

The following screenshots demonstrate a sample interaction with the Smart Board. You do not have to follow this precisely and are encouraged to come up with your own designs. These illustrates the required functionality.

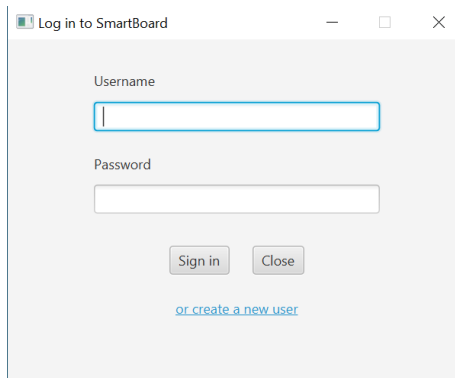


Figure 1. Login window.

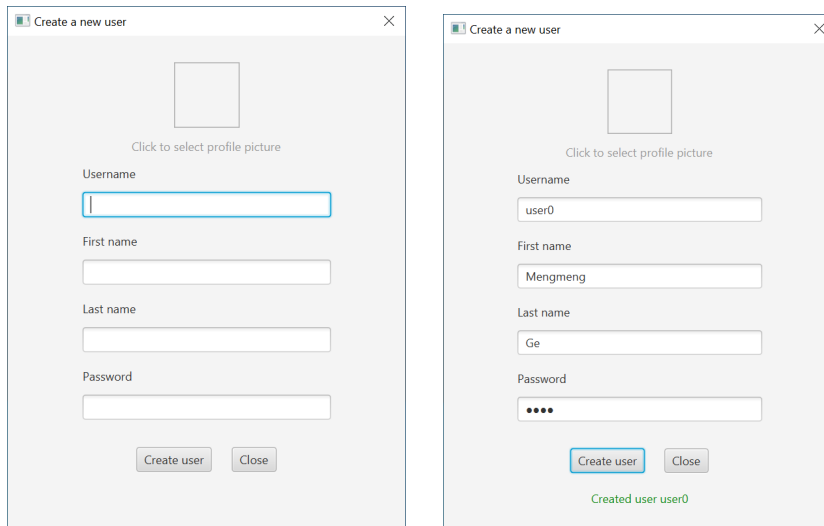


Figure 2. a) Profile creation window with empty information. b) Profile creation window after the user fills in the information and clicks “Create user” button.

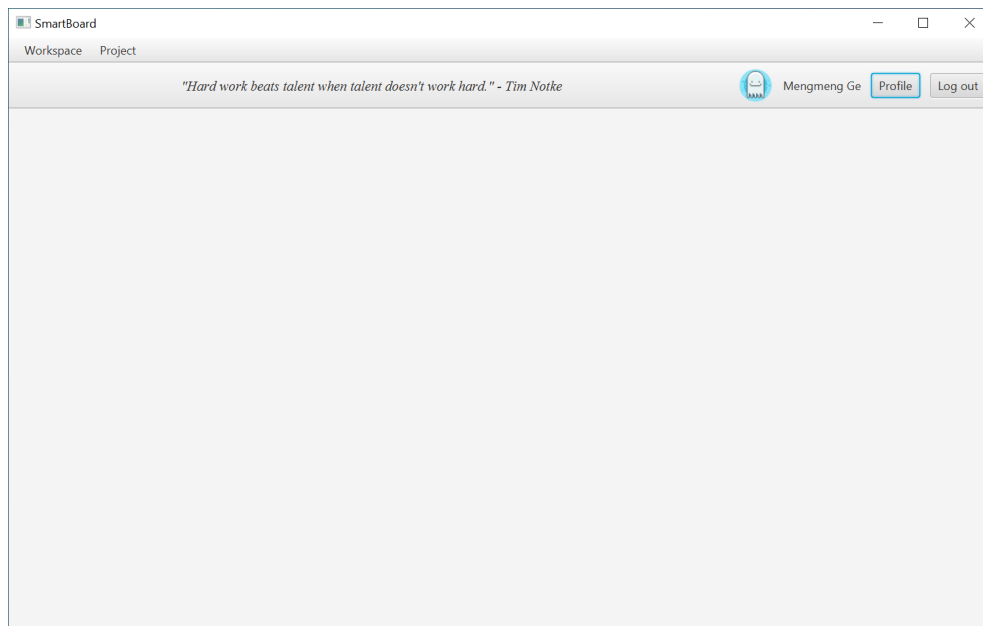


Figure 3. Empty workspace after the user first logs in.



Figure 4. Profile editing window.

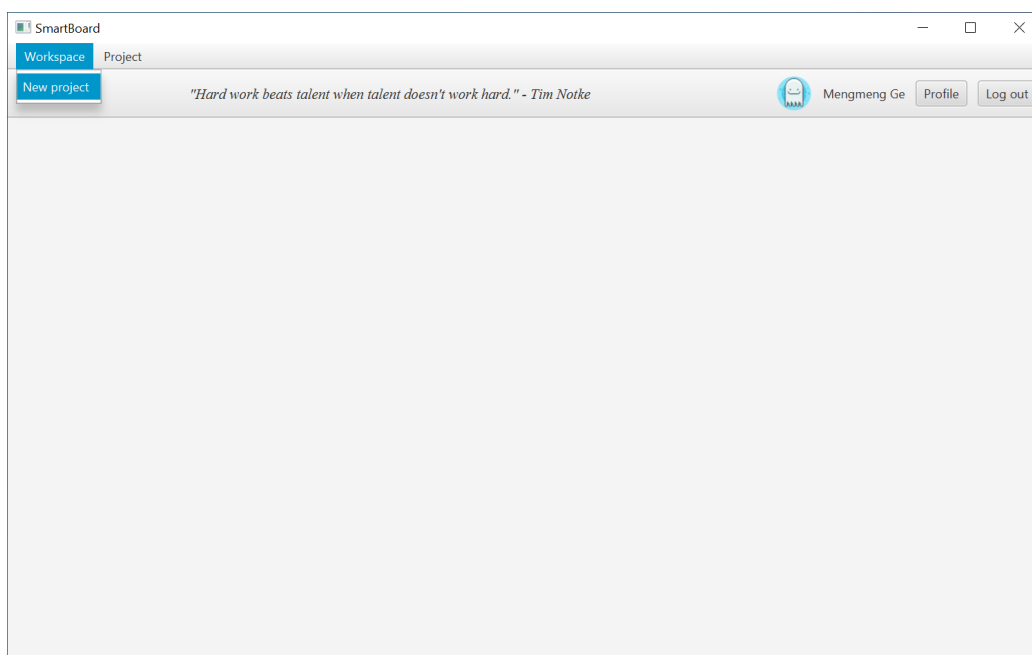


Figure 5. Menu for

workspace.

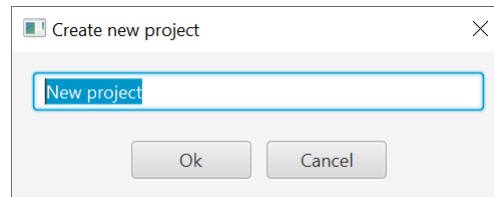


Figure. 6 Create a new project.

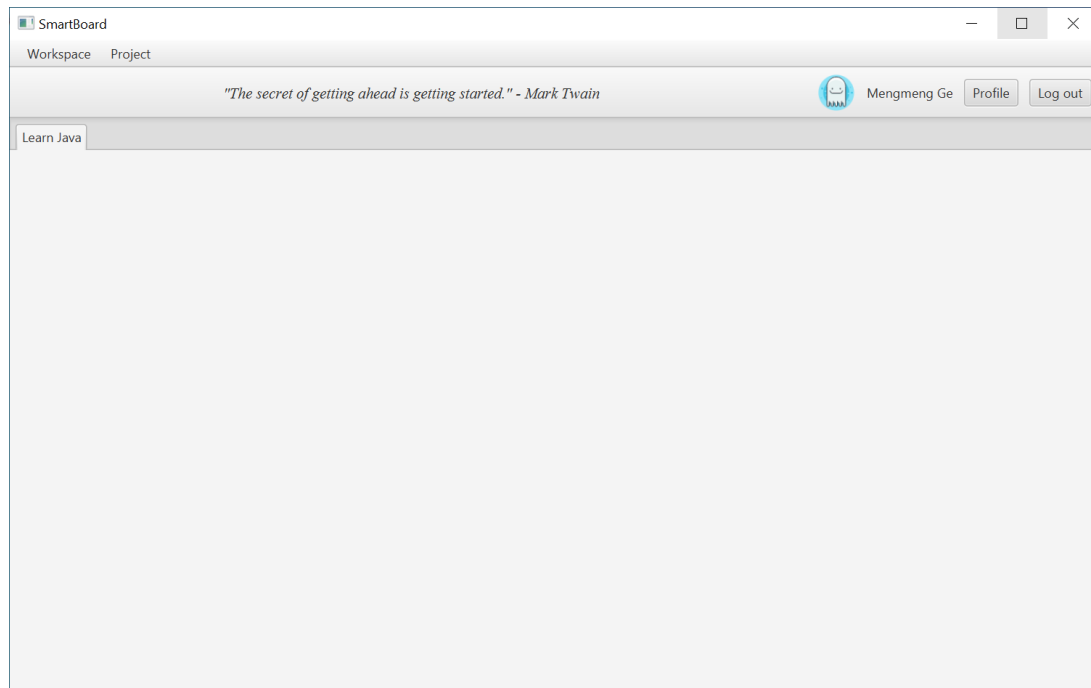


Figure 7. A new project created on the workspace.

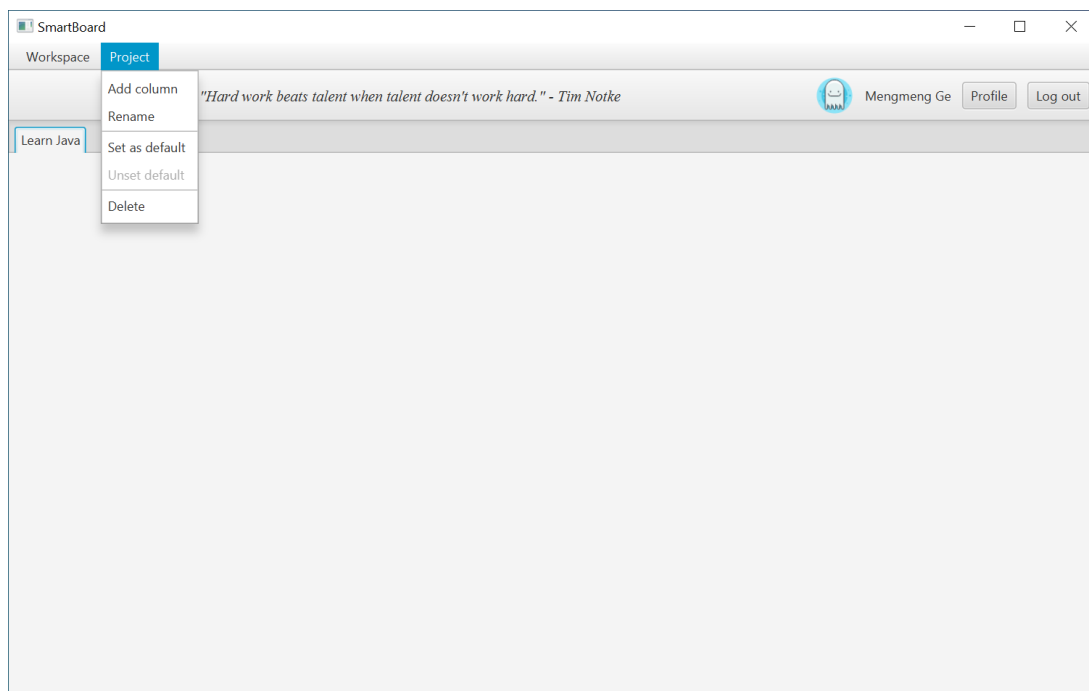


Figure 8. Menu for project.

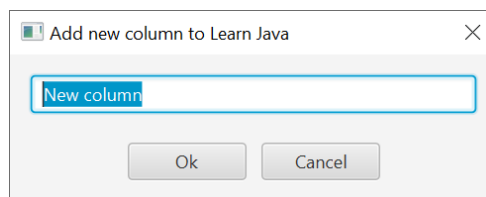


Figure 9. Add a new column on a project.

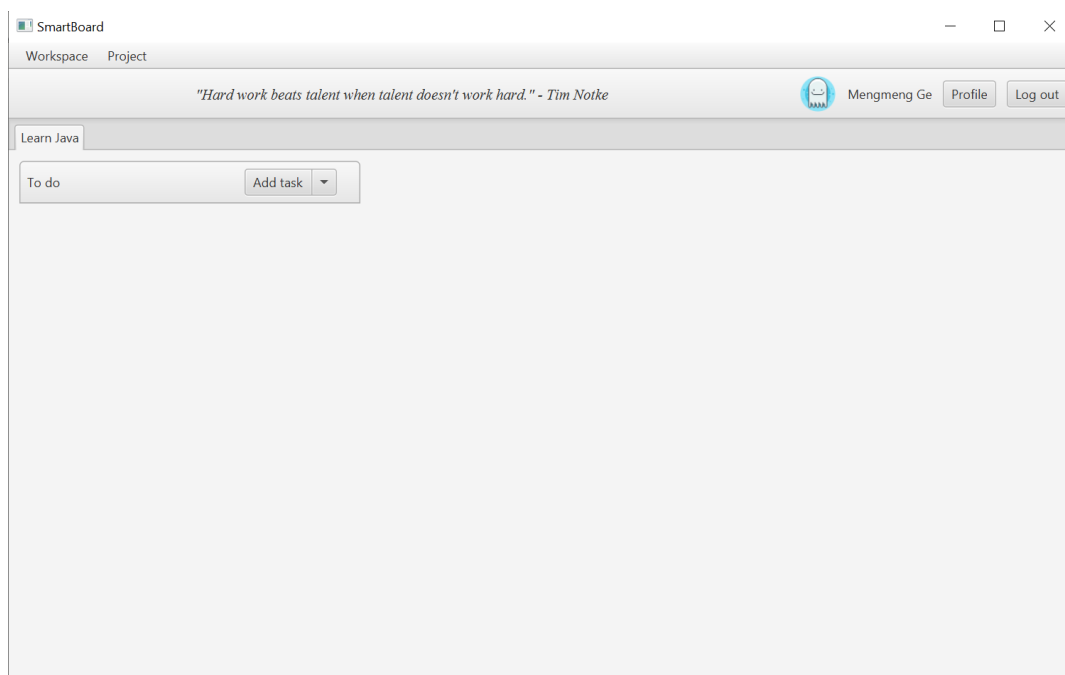


Figure 10. A new column

created on the project board.

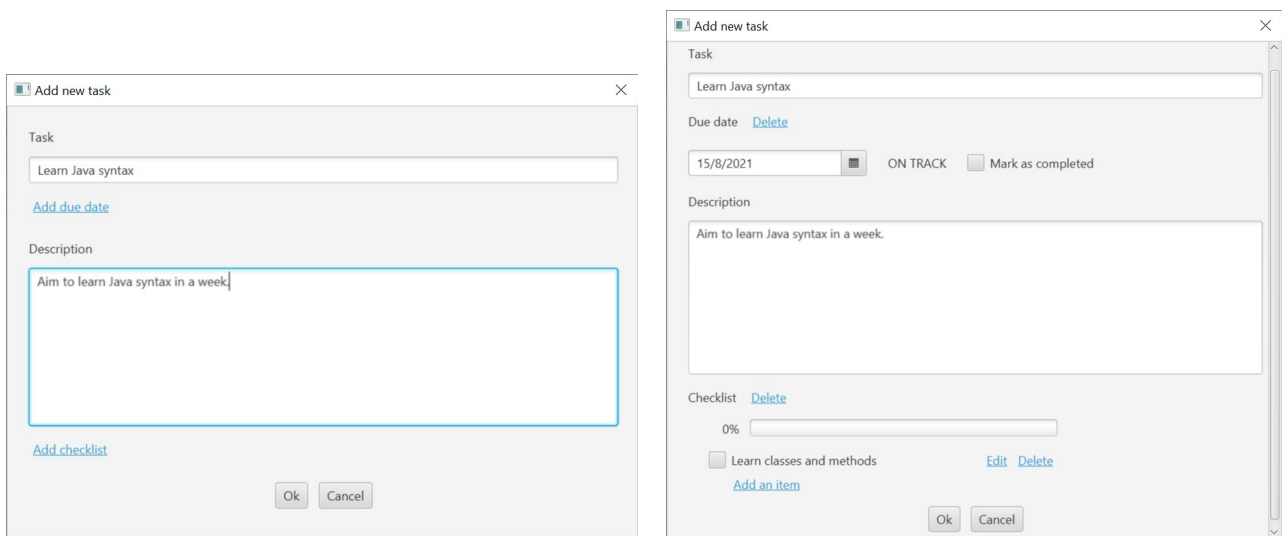


Figure 11. a) Task editor window with name and description; b) Task editor window with name, due date, description and checklist.

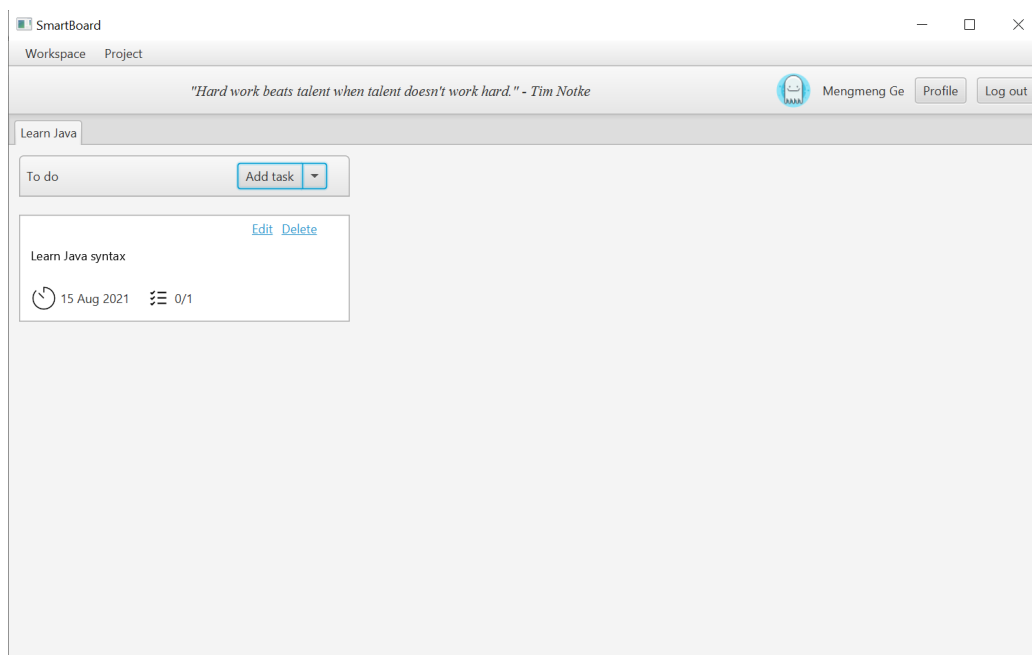


Figure 12. Task card shown on the project board.

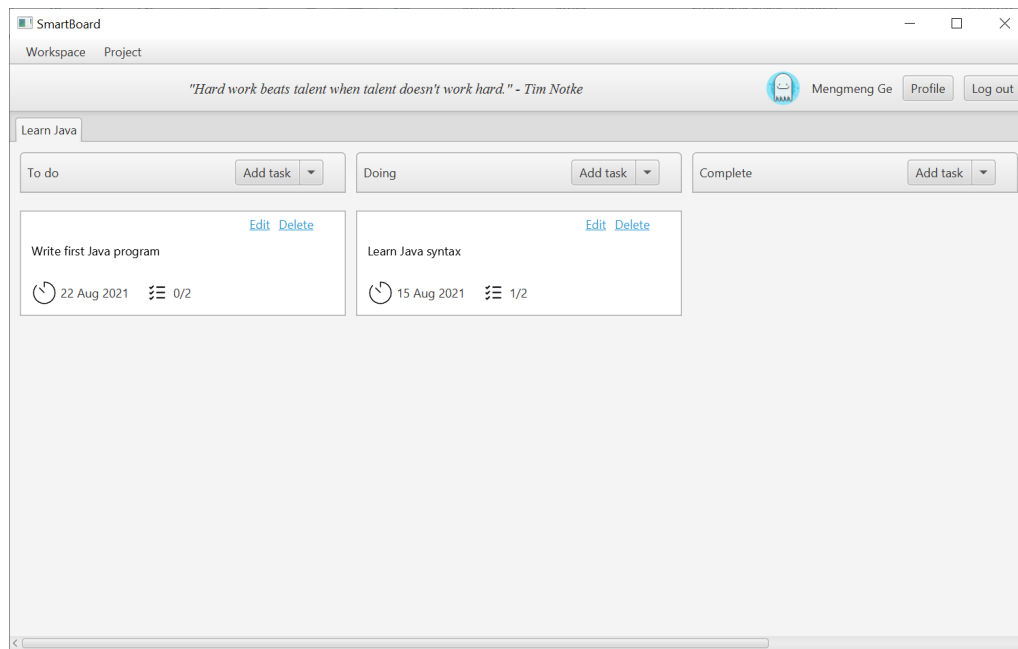


Figure 13. Add multiple columns and multiple tasks.

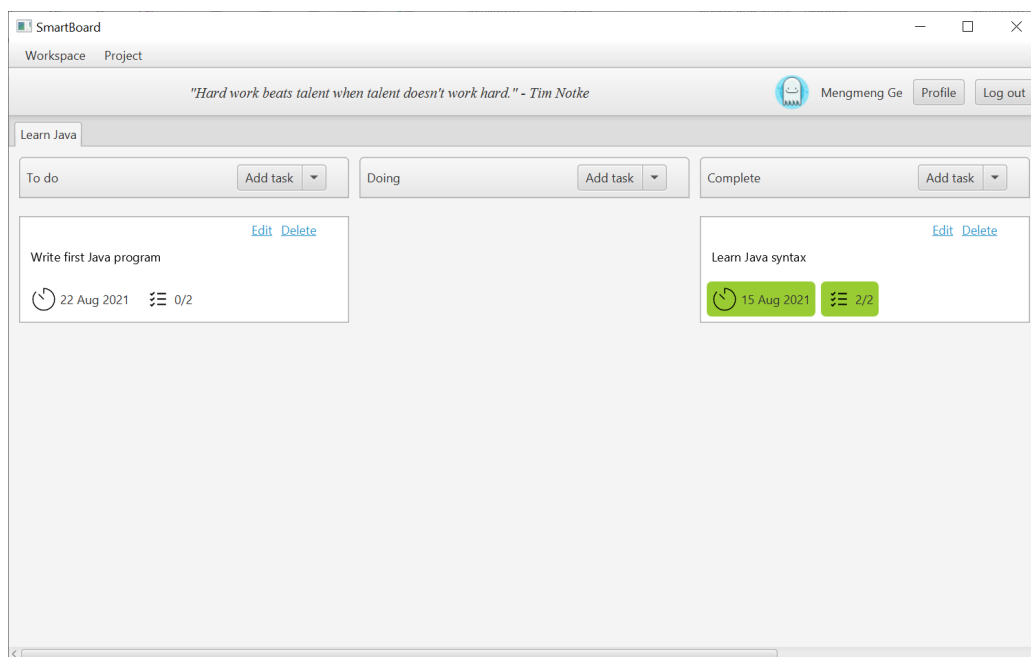


Figure 14. One task is moved from Doing column to Complete column.

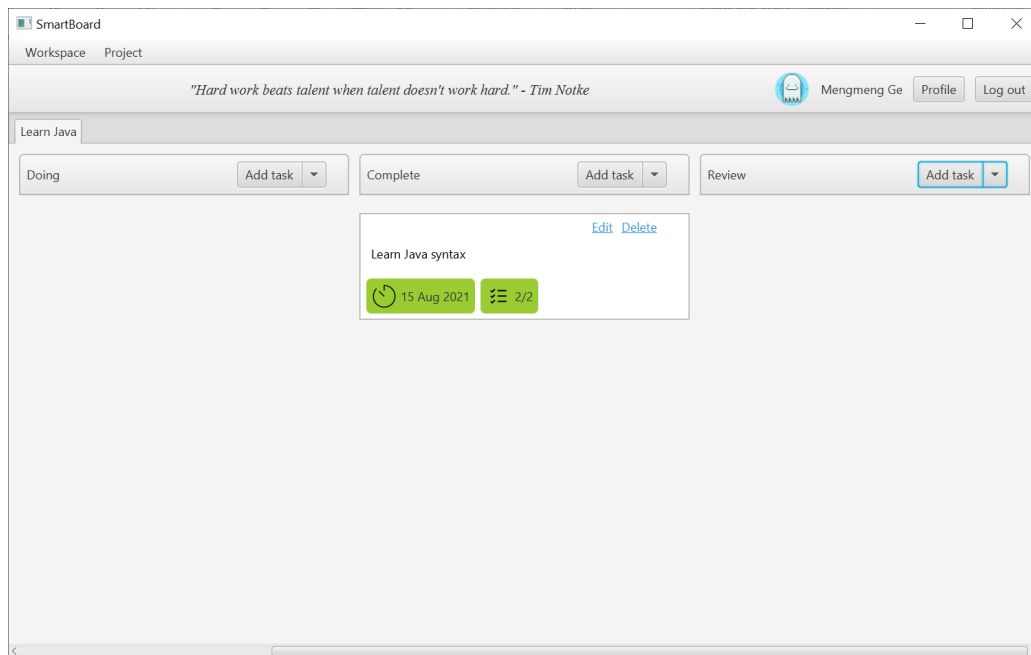


Figure 15. Scroll bar shown on the board to enable viewing of all columns.

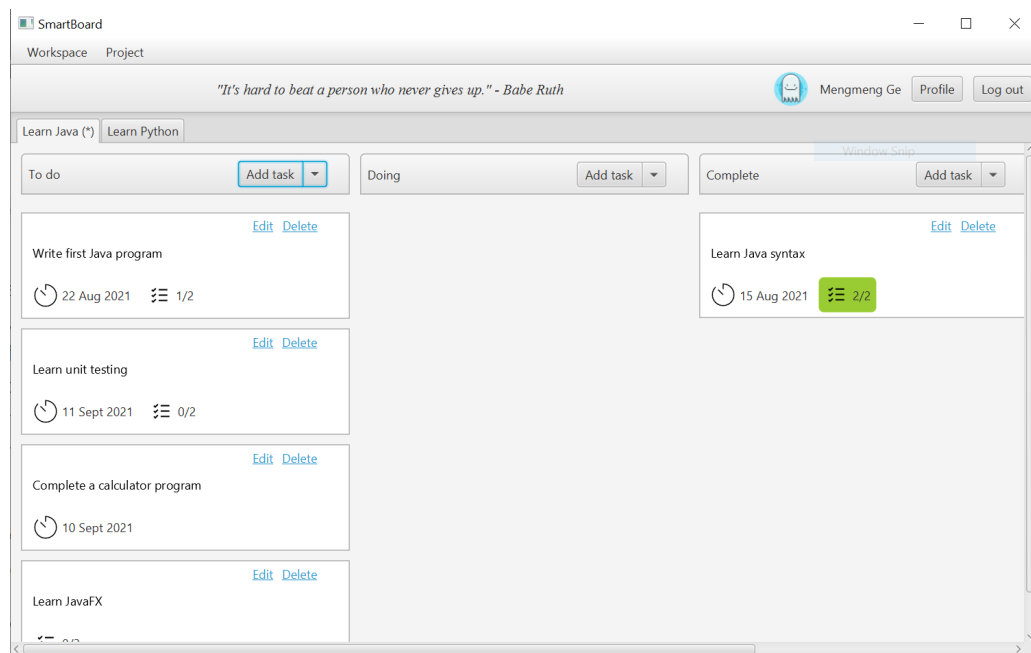


Figure 16. Scroll bar shown on the board to enable viewing of all tasks within each column.

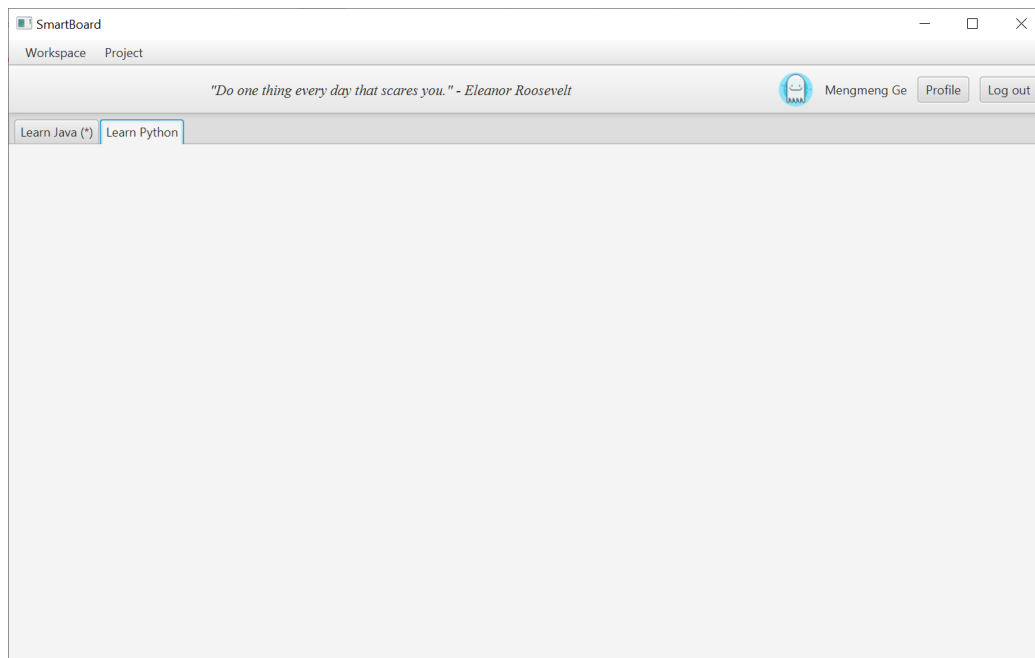


Figure 17. Add a new project.

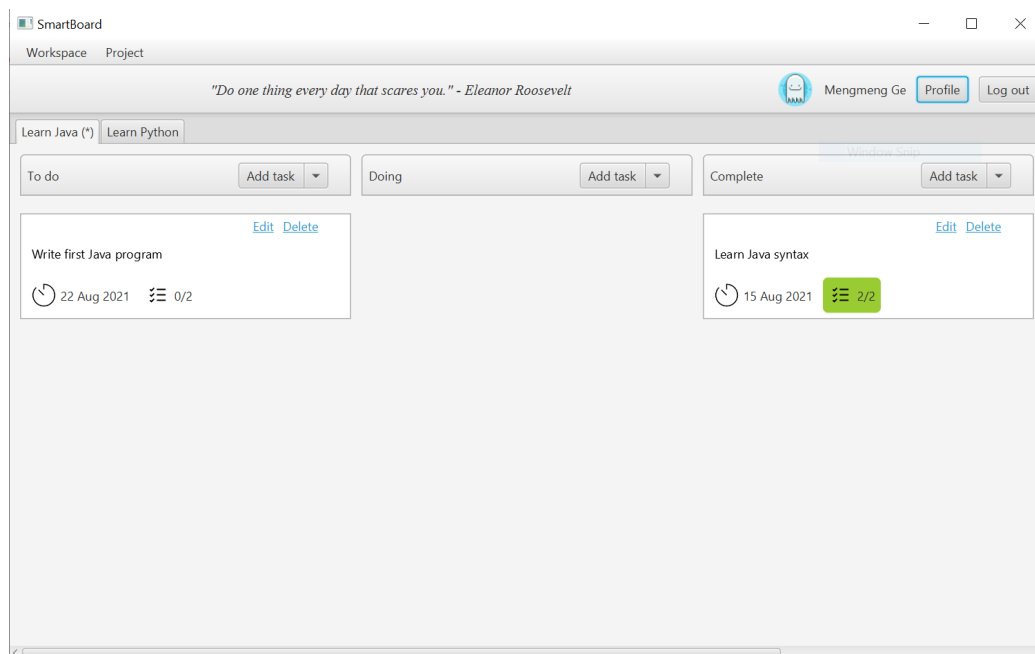


Figure 18. Set Learn Java as the default project (* indicated the default project).

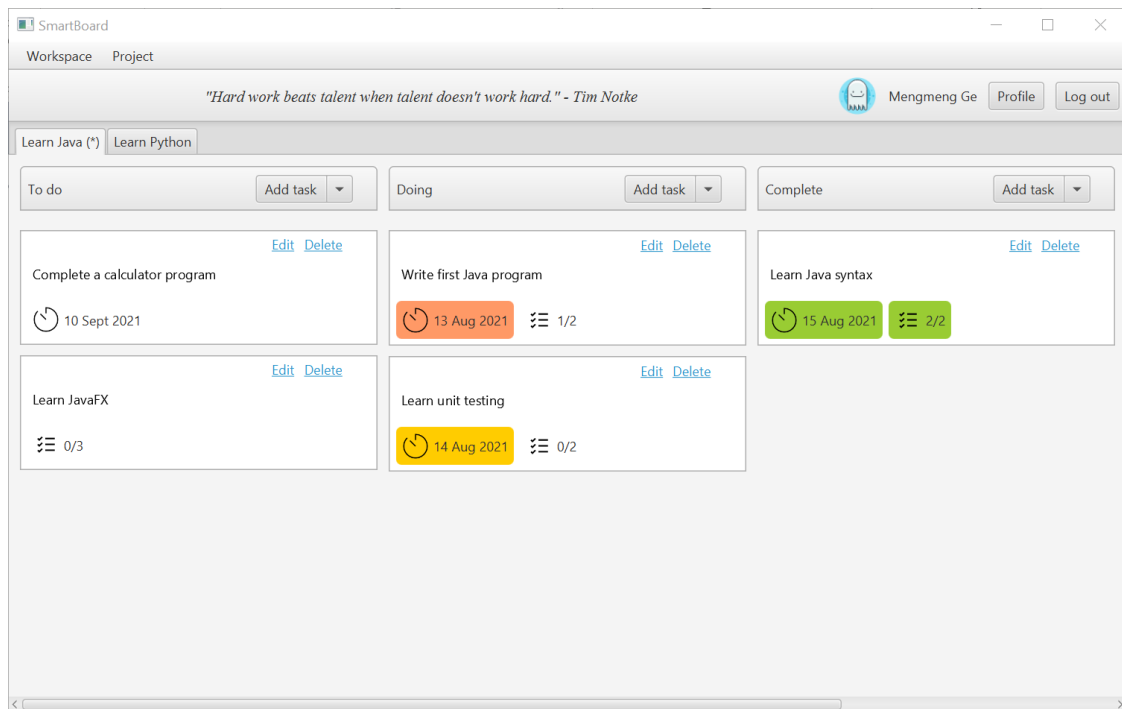


Figure 19. Colour indicator (yellow when the due date is approaching; orange when overdue; green when the task is complete within the due date).