```
/**
 *
 */
package projects;

import java.math.BigDecimal;
import java.util.List;
import java.util.Objects;
import java.util.Scanner;

import projects.dao.exception.DbException;
import projects.entity.Project;
import projects.service.ProjectService;

/**
 * @author Candace Samuels
 */

public class ProjectsApp {
	private Scanner scanner = new Scanner(System.in);
	private ProjectService projectService = new ProjectService();
	private Project curProject;

	//@formatter:off
	private List<String> operations = List.of(
				"1) Add a project",
				"2) List projects",
				"3) Select a project"
				);
```

```
	//@formatter:on

	public static void main(String[] args) {
		new ProjectsApp().processUserSelections();
	}

	private void processUserSelections() {
		boolean done = false;
		while (!done) {
			try {
				int selection = getUserSelection();

				switch(selection) {
				case -1:
					done = exitMenu();
					break;
				case 1:
					createProject();
					break;
				case 2:
					listProjects();
					break;
				case 3:
					selectProject();
					break;

				default:
					System.out.println("\n" + selection + " is not a valid selection.
Try again.");
```

```
                        }
                    } catch (Exception e) {
                        System.out.println("\nError: " + e + " Try again.");
                    }
                }
            }


// WEEK 10: START
            private void selectProject() {
                listProjects();
                Integer projectId= getIntInput("Enter a project ID to select a project");

                /* Unselect the current project. */
                curProject = null;

                /* This will throw an exception if an invalid project ID is entered. */
                curProject = projectService.fetchProjectByProjectId(projectId);
}

            private void listProjects() {
                List<Project> projects = projectService.fetchAllProjects();

                System.out.println("\nProjects:");

                    projects.forEach (
                      project -> System.out.println(" " + project.getProjectId() + ":" +
project.getProjectName()));
                        // Lambda expression used
```

```
                }

// WEEK 9 START
        @SuppressWarnings("unused")
        private void createProject() {
                String projectName = getStringInput("Enter the project name");
                BigDecimal estimatedHours = getDecimalInput("Enter the estimated hours");
                BigDecimal actualHours = getDecimalInput("Enter the actual hours");
                Integer difficulty = getIntInput("Enter the project difficulty (1-5)");
                String notes = getStringInput("Enter the project notes");

                Project project = new Project();

                project.setProjectName(projectName);
                project.setEstimatedHours(estimatedHours);
                project.setActualHours(actualHours);
                project.setDifficulty(difficulty);
                project.setNotes(notes);

                Project dbProject = projectService.addProject(project);
                System.out.println("You have successfully created project: " + dbProject);
        }

        private BigDecimal getDecimalInput(String prompt) {
                String input = getStringInput(prompt);

                if (Objects.isNull(input)) {
                        return null;
                }
```

```
            try {
                    return new BigDecimal(input).setScale(2);
            }
            catch (NumberFormatException e) {
                    throw new DbException(input + " is not a valid number.");
            }
}

private boolean exitMenu() {
            System.out.println("Exiting the menu.");
                    return true;
}

private int getUserSelection() {
            printOperations();

            Integer input = getIntInput("Enter a menu selection");
                    return Objects.isNull(input) ? -1 : input;
}

private Integer getIntInput(String prompt) {
            String input = getStringInput(prompt);

            if (Objects.isNull(input)) {
                    return null;
            }
            try {
                    return Integer.valueOf(input);
```

```
			}
			catch (NumberFormatException e) {
				throw new DbException(input + " is not a valid number.");
			}
		}

		private String getStringInput(String prompt) {
			System.out.print(prompt + ": "); // test the application
			String input = scanner.nextLine();

			return input.isBlank() ? null : input.trim();
		}

		private void printOperations() {
			System.out.println("These are the avaliable menu sections. Press the Enter key to
quit:");
		// WEEK 10 START
			if(Objects.isNull(curProject)) {
				System.out.println("\nYou are not working with a project");
			}
			else {
				System.out.println("\nYou are working with project: " + curProject);
			}
		// WEEK 10 END
			operations.forEach(line -> System.out.println("  " + line));
		}
}
//WEEK 9: END
```