# CSC207 PROJECT REPORT TEMPLATE

FINAL Project Report

*Created 2022-12-06*

Bamboggled

**Hassan El-Sheikha**

**Mustafa Zafar**

**Sultan Al-Dhalaan**

**Kevin John Thevara**

LINK TO DEMO VIDEO: https://utoronto-my.sharepoint.com/:v:/g/personal/hassan\_elsheikha\_mail\_utoronto\_ca/EabxASCoe35DtXgbndBPFy0BZvdsqFl8ni4ZlsjjVfbizg?e=nRDVhM

# Section 1: REPORT SUMMARY

Boggle is a game that dates back to the 1970s. Back then, it was much easier to play similar games on a physical board, but nowadays the trend has been to play such games digitally, such as on a computer or smartphone. However, not everyone enjoys a text-based version of a game; it’s too boring, and the black-and-white (or black-and-green) color scheme is irritating to the eye. So, the goal of this project is to create an upgraded GUI version of Boggle, with many upgrades to the initial rules of the game for fun.

# Section 2: PROCESS DOCUMENTATION

***2.1. SPRINT 1 OVERVIEW***

**2.1.1 Sprint Overview:**

Before starting this sprint, there were revisions to the design document. We decided to explicitly create a model for handling our business logic as well as view classes (with controllers embedded). The first task was to first create an interface for the model, so that the team members working on frontend could be able to design their classes without worrying about the model being implemented. After this was complete, the members working on backend were to implement the model, and the members working on frontend implemented the views (and event handlers). Furthermore, to facilitate our work and avoid unexpected errors, we used continuous integration (using Maven) for this project, and this was to be configured during this sprint.

**2.1.2 Stories Selected for this Sprint:**

For this sprint, we decided to select stories 0.0 through 1.3.

**2.1.3 Team Capacity:**

We expect to be able to complete the components of the Leaderboard by Thursday, November 22.

**2.1.4 Participants:**

Hassan:

* Scrum Master; Hold Daily Meetings and Monitor Progress
* 0.0 Setup Continuous Integration
* 1.1.0 Model Interface
* 1.1.1 Model Implementation (partial)

Mustafa:

* 1.3 Create views and event handlers for BoardView 5x5 and 4x4, and GameEndView.

Kevin:

* 1.3 Create views and event handlers for WelcomeView, InstructionsView, and PlayerInitView.

Sultan:

* 1.1.1 Model Implementation (partial)
* 1.1.2 Text-Based Version
* 1.1.3 Model Testing

**2.1.5 Tasks Completed:**

Everything was finished, except for Model Testing. Hassan ended up completing all of Sultan’s other tasks as Sultan did not contribute during this sprint.

***2.2. SPRINT 1 PRODUCT BACKLOG***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | ID | Owner | Description | Implementation Details | Priority | Effort |
| Setup Continuous Integration | 0.0 | Hassan | As a developer doing a code review on another team member’s work, I want to first check that the code compiles, without compiling it on my machine. I also shouldn’t be allowed to merge a branch where the code on it fails any tests. | Use Maven and a CI platform like CircleCi or TravisCi to implement this. | 1 | 1 |
| Model Interface | 1.1.0 | Hassan | As a frontend developer working on the view, I want to be able to reference and call methods of the model in my classes, regardless if it has been implemented or not. | Create an interface IBoggleModel. This will have methods to start a new game, end a game, get and add to the current word, submit the current word, etc. | 3 | 1 |
| Model Implementation | 1.1.1 | Hassan, Sultan (actually implemented by Hassan) | This is the implementation for the interface described above. | -Implement all methods in the IBoggleModel interface in a concrete BoggleModel class.  -Create algorithms for updating the paths leading to a certain word (there could be multiple on the board!).  -Create | 3 | 3 |
| Model Test-Suite | 1.1.2 | Sultan | As a developer working on the BoggleModel, I want to have a high-quality test suite to run tests against my code so that I can make sure it is correct. | Create tests for algorithms and operations in the BoggleModel. | 1 | 1 |
| Text-Based Version | 1.1.5 | Sultan (Actually implemented by Hassan) | As a physical board game “boggler,” I want to be able to play with friends on my computer so that I don’t have to manually check for valid words and memorize scores.  This text-based version of boggle should accept letter-by-letter inputs instead of whole words, so that it simulates the actual GUI version. | Create a TextBased view that accepts character-by-character input instead of whole words. This should update the board to reflect the current path the user has selected after every character input. | 1 | 2 |
| GUI | 1.3 | Mustafa, Kevin, Hassan | As a boggle user, I want the board to be displayed using a GUI so that it can be easier for me to play the game (as opposed to the text-based version). This new GUI should still allow me to play either with a friend or against the computer. This version will be playable with only a keyboard. | Create 5 views:  -The WelcomeView, which is like a home page for the game.  -The  -InstructionsView, which is a page containing instructions on how to play the game  -PlayerInitView, which is a page where the user configures the board size and inputs player names.  -4x4 and 5x5 BoardView. This is the main game page, where the user selects words and increases their score.  -EndGameView, where the winner’s name is displayed and the user can opt to play again. | 3 | 3 |

***2.3. SPRINT 1 CODE REVIEWS***

We’re expecting that each team member will make some changes to the team repository at each sprint (meaning we expect to see roughly weekly commits). Moreover, we’re expecting that before changes on feature branches are transferred to your team’s develop branch, that your team will conduct code reviews. Each team member should provide at least one code review for one of their peers at each sprint iteration. Your reviews will be documented in your repository, but we ask that your briefly document them here as well using this format:

|  |  |  |
| --- | --- | --- |
| **Story Reviewed** | **Name of Reviewer** | **Pull Request Link** |
| [DEV-1.1] Create model package and wordlist | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/15 |
| [DEV-1.1] Create BoggleDictionary class | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/16 |
| [DEV-1.1] Create BoggleGrid class | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/17 |
| [DEV-1.1] Create die and random board string generator | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/18 |
| [DEV-1.1] Create path class | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/19 |
| [DEV-1.1] Create PossiblePathContainer class | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/20 |
| [DEV-1.1] create letter by letter path search algorithm | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/21 |
| [DEV-1.1] Create Player class | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/23 |
| [DEV-1.1] Create model interface | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/24 |
| [DEV-1.1] Implement model | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/25 |
| [DEV-1.1] Create text based view | N/A (Pull request was in depreciated repository) | https://github.com/Mind-Bogglers/Bamboggled/pull/26 |
| [DEV-N/A] Update pom file to work with javafx | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/41 |
| [DEV-1.3] create scene builder graph for board view and game control | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/42 |
| [DEV-1.3] scene builder graphs | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/47 |
| [DEV-1.3] Singleton implementation for model | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/49 |
| [DEV-1.3] fix integration fxml model | Kevin | https://github.com/Mind-Bogglers/Bamboggled/pull/50 |
| [DEV-1.3] Bug fixes | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/51 |
| [DEV-1.3] connect game config view with board view | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/53 |
| [DEV-1.3] Single player initialization fix | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/54 |
| [DEV-1.3] Event handling for BoardGameView | Kevin | https://github.com/Mind-Bogglers/Bamboggled/pull/55 |
| [DEV-1.3] create scene builder graph for game end view | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/56 |

***2.4 SPRINT 1 RETROSPECTIVE***

When a sprint is completed, hold a retrospective meeting and ask one team member to take notes. Place a short record of each retrospective meeting in this section! The details should include:

* Participants Hassan, Kevin, Mustafa, Sultan
* Things that are not complete: Testing for model, 5x5 Board
* Things that went well: (Almost) all user stories were completed, and teamwork was great at the latter half of the sprint. Also, the product was fully functional and appealing.
* Things that need improvement: Not everyone contributed, Hassan ended up having to take many tasks that he didn’t originally own so that the sprint could end on time (even though the sprint ended late).
* A summary of any bad practices that will not be repeated moving forward: Every team member must participate.
* Your team’s best/worst experience during this sprint: Finishing the BoardView was a great experience, since it allowed us to see our product in action. Figuring out a workaround to a limitation of JavaFX took us a lot of time, so this was the worst experience.

***2.5. SPRINT 2 OVERVIEW***

**2.5.1 Sprint Overview:**

From the last sprint, Sultan still needed to implement testing for the model. Also, more tasks we shifted over to Sultan for this week, as the rest of the team did more work than originally planned in the previous sprint. For this sprint, the undo feature was to be implemented, as well as the visually impaired mode.

**2.5.2 Stories Selected for this Sprint:**

For this sprint, we decided to select stories 2.1 and 2.2.

**2.5.3 Team Capacity:**

We expect to be able to complete the components of the Leaderboard by December 3.

**2.5.4 Participants:**

Hassan:

* Scrum Master; Hold Daily Meetings and Monitor Progress
* 2.1 Implement memento pattern for the BoggleModel class and implement undo feature for BoardView.

Mustafa:

* Implement BoardView 5x5, which wasn’t implemented in the last sprint.

Kevin:

* 2.2.1 Audio-Only Mode

Sultan:

* 2.2.0 Audio-Related Algorithm in Model
* 2.2.1 Audio-Only Mode
* 1.1.3 Model Testing (from last week)

**2.5.5 Tasks Completed:**

Everything was finished, except for 2.2.0 Audio-Related Algorithm in Model and the audio-only mode for the BoardView. Audio-only mode was implemented everywhere else in the application, however. Also, the 5x5 SceneBuilder graph but never implemented due to time constraints.

***2.6. SPRINT 2 PRODUCT BACKLOG***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | ID | Owner | Description | Implementation Details | Priority | Effort |
| Audio-Related Algorithm in Model | 2.2.0 | Sultan (Never Implemented) | As a visually impaired Bamboggled player, I want the screen to read out my next possible moves after I select a letter. | Loop over all possible paths and fetch possible next moves. | 3 | 1 |
| Audio-  Only Mode | 2.2.1 | Kevin, Sultan (Actually implemented by Kevin—Incomplete) | As a visually impaired person, I want a UI that allows me to interact with the board without visual elements, but rather only with audio so that I can play the game in the same way that non-visually impaired people play. This new audio-only mode should allow me to play against the computer. | Create button or shortcut that activates visually impaired mode. If this option is enabled, the computer will read out information on the current view. | 2 | 3 |
| Undo | 2.1 | Hassan, Mustafa (actually implemented by Hassan). | As a Boggle player who makes errors when choosing words, I don’t want to have to restart choosing my whole word. Instead, I want to press backspace to delete the most recently pressed letter. | -Memento implementation for model  -Event handling for backspace key | 3 | 2 |
| Text-Based Version | 1.1.5 | Sultan (Actually implemented by Hassan) | As a physical board game “boggler,” I want to be able to play with friends on my computer so that I don’t have to manually check for valid words and memorize scores.  This text-based version of boggle should accept letter-by-letter inputs instead of whole words, so that it simulates the actual GUI version. | Create a TextBased view that accepts character-by-character input instead of whole words. This should update the board to reflect the current path the user has selected after every character input. | 1 | 2 |
| GUI | 1.3 | Mustafa | As a boggle user, I want the board to be displayed using a GUI so that it can be easier for me to play the game (as opposed to the text-based version). This new GUI should still allow me to play either with a friend or against the computer. This version will be playable with only a keyboard. | 5x5 board view | 3 | 3 |

***2.7. SPRINT 2 CODE REVIEWS***

We’re expecting that each team member will make some changes to the team repository at each sprint (meaning we expect to see roughly weekly commits). Moreover, we’re expecting that before changes on feature branches are transferred to your team’s develop branch, that your team will conduct code reviews. Each team member should provide at least one code review for one of their peers at each sprint iteration. Your reviews will be documented in your repository, but we ask that your briefly document them here as well using this format:

|  |  |  |
| --- | --- | --- |
| **Story Reviewed** | **Name of Reviewer** | **Pull Request Link** |
| [DEV-2.2] undo feature | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/65 |
| [DEV-2.1] find text-to-speech api | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/67 |
| [DEV-1.1] unit test for model | Hasan | https://github.com/Mind-Bogglers/Bamboggled/pull/68 |
| [DEV-0.0] configure maven to build executable jar with dependencies | Mustafa | https://github.com/Mind-Bogglers/Bamboggled/pull/69 |
| [DEV-2.1] blind mode for player selection frontend | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/70 |
| [DEV-2.3] Created 5x5 boggle board view | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/72 |
| [DEV-2.1] refactoring | Kevin | https://github.com/Mind-Bogglers/Bamboggled/pull/73 |

***2.8 SPRINT 2 RETROSPECTIVE***

When a sprint is completed, hold a retrospective meeting and ask one team member to take notes. Place a short record of each retrospective meeting in this section! The details should include:

* Participants: Hassan, Kevin, Mustafa
* Things that are not complete: 5x5 BoardView implementation, audio-related algorithm in model, audio-only mode for BoardView.
* Things that went well: (Almost) all user stories were completed, and teamwork was great throughout the sprint.
* Things that need improvement: Not all tasks were completed. Kevin and Hassan had to take on extra responsibilities on frontend.
* A summary of any bad practices that will not be repeated moving forward: Every team member must participate throughout the sprint and not just do one of their tasks for the week then stop. Also, attributes that shouldn’t belong in a class will never be thrown into that class just so that we can have someplace to store it (this occurred when a visually impaired mode indicator attribute was added to the model class, though the model has nothing to do with visually-impaired mode).
* Your team’s best/worst experience during this sprint: Getting the undo feature to work was the best experience. The worst experience was handing key events pertaining to the visually-impaired mode.

***2.9. SPRINT 2 OVERVIEW***

**2.9.1 Sprint Overview:**

This sprint focused on proper documentation of code and finalizing the project.

**2.9.2 Stories Selected for this Sprint:**

For this sprint, we selected stories 3.1 and 3.2.

**2.9.3 Team Capacity:**

We expect to be able to complete the components of the Leaderboard by December 8.

**2.9.4 Participants:**

Hassan:

* Scrum Master; Hold Daily Meetings and Monitor Progress
* Finalize final report and shoot demo video
* Record implementation details
* Draw UML for Memento and Singleton

Mustafa:

* Draw UML for observer

Kevin:

* Document views
* Draw UML for views

Sultan:

**2.9.5 Tasks Completed:**

Everything was completed.

***2.10. SPRINT 3 PRODUCT BACKLOG***

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | ID | Owner | Description | Implementation Details | Priority | Effort |
| Final Report and Recording | 3.1 | All | Write and finalize final report, and shoot video. | Write and finalize final report, and shoot video | 2 | 3 |
| Documentation of view | 3.2 | Kevin | Document all view classes and their methods. | Document all view classes and their methods. | 2 | 2 |

***2.11. SPRINT 3 CODE REVIEWS***

We’re expecting that each team member will make some changes to the team repository at each sprint (meaning we expect to see roughly weekly commits). Moreover, we’re expecting that before changes on feature branches are transferred to your team’s develop branch, that your team will conduct code reviews. Each team member should provide at least one code review for one of their peers at each sprint iteration. Your reviews will be documented in your repository, but we ask that your briefly document them here as well using this format:

|  |  |  |
| --- | --- | --- |
| **Story Reviewed** | **Name of Reviewer** | **Pull Request Link** |
| [DEV-3.2] documenting code | Hassan | https://github.com/Mind-Bogglers/Bamboggled/pull/79 |
| [DEV-3.1] final report and design document updates | Kevin | https://github.com/Mind-Bogglers/Bamboggled/issues/74 |

***2.12 SPRINT 3 RETROSPECTIVE***

When a sprint is completed, hold a retrospective meeting and ask one team member to take notes. Place a short record of each retrospective meeting in this section! The details should include:

* Participants: Hassan, Kevin, Mustafa
* Things that are not complete: N/A
* Things that went well: We finished on time! And with a working product!

# Section 3: SUMMARY

In this final section, briefly summarize both your project **accomplishments** and its **limitations**.

This project went well overall and was a great experience. The product we finished did not include everything stated in the original design document, since it lacks powerups and a timed mode. However, this was because the expected number of sprints was cut from 4 to 3, and the last sprint was more focused on documentation and cleanup. Our design changed since we decided to use MVC, implementing a BoggleModel class for the model and view classes (with controllers embedded) for each view of the application. Also, an unexpected JavaFX limitation required that we implement the singleton pattern for the BoggleModel class, which was not something we initially planned on doing. Another limitation we had was that some team members had to take on additional responsibilities that they did not initially get assigned to, due to other team members not completing their own tasks. This is why some user stories ended up being implemented by different group members than initially proposed. Additionally, the location that the bridge pattern was (supposed to be) used in changed. However, the experience was great, and we really enjoyed implementing the game (not more than we enjoyed playing it though)!