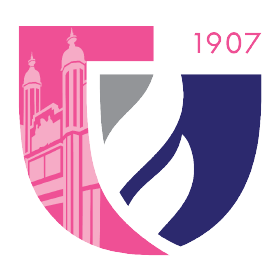
**CENTRO ESCOLAR UNIVERSITY**



SINGING BEE

**CUNANAN, GIVEN V.**

**ESPINOZA, ARVIN S.**

**FRANCISCO, EARL ACE DOMAND F.**

**LUMANOG, IAN M.**

**MANALO, ALTHEA IRISH M.**

**MENDOZA, JAOSCH NATHAN C.**

**TOMATE, ALEXENE FAITH S.**

Adviser:

**Dave B. Mercado**

**Centro Escolar University**

School of Science and Technology

Date of Submission:

January 2023

**ACKNOWLEDGEMENT**

We deliver the sincerest gratitudes to everyone that helped this project come to fruition. The meaningful endeavors we experienced would not have been possible without the people who steered us to the path of development.

To God, who blessed us with the intellectual capacity to administer this project. His guidance helped us in producing the best outcomes we can muster to the best of our God-given abilities.

To our Algorithm adviser, Mr. Dave Mercado, we would like to extend our warmest and heartfelt gratitude for continually guiding us in the creation of this project as well as for helping us and supporting us all throughout the completion of this project. We thank you.

To our mock panelists, who gave us a precious portion of their time to give valuable feedback for the improvement of our presentations.

Finally, we would like to give our thanks to the friends and families who cheered us on throughout our project-building phases, your suggestions are highly appreciated and has greatly helped in selecting songs and features to include in the program.

**APPROVAL SHEET**

This is to certify that this project entitled **Singing Bee** prepared and submitted by Given V. Cunanan, Arvin S. Espinoza, Earl Ace Domand F. Francisco, Ian M. Lumanog, Althea Irish M. Manalo, Jaosch Nathan C. Mendoza, and Alexene Faith S. Tomate in partial fulfillment for the requirements of Algorithm and Logic Formulation is hereby accepted.

**DAVE B. MERCADO**

Subject Adviser

**TABLE OF CONTENTS**

Page

TITLE PAGE ………………………………………………………… i

ACKNOWLEDGEMENT ………………………………………………………… ii

APPROVAL SHEET ………………………………………………………… iii

INTRODUCTION ………………………………………………………… 1

Background of the Study ………………………………………………………… 1

Significance of the Study ………………………………………………………… 1

Objectives ………………………………………………………… 2

METHODOLOGY ………………………………………………………… 3

Program Design ………………………………………………………… 3

Program Flow ………………………………………………………… 4

Work Flow ………………………………………………………… 5

Division of Labor …....…………………………………………………… 6

Tools and Platforms Used ………………………………………………………… 7

CONCLUSION ………………………………………………………… 8

Project Timeline ………………………………………………………… 8

Results ………………………………………………………… 9

Challenges & Resolutions ………………………………………………………… 9

Recommendations ………………………………………………………… 11

**Chapter 1**

**INTRODUCTION**

**Background**

One of the most memorable Filipino variety game shows from the past is The Singing Bee. It was originally an American game show of the same name, which first aired on NBC in the early 2007’s. Later on, the Philippines adapted it whilst keeping the name, and the show finally aired locally under ABS-CBN on April 21, 2008. The Singing Bee is based on karaoke culture where contestants sing along while guessing the lyrics of timeless songs in a bee-themed competition. However, the show has long been abolished.

With longing for the simpler days that have passed, the developers of this programming project asked themselves, “What if we immortalize this old, memorable TV show by turning it into a computer game?” The team decided to push through and take note of the game’s core mechanics. In the TV show, the player must listen to a snippet of a certain song and supply the missing words by singing along. However, due to time restrictions in project-building, the developers weren’t able to add a speech-to-text feature. Hence, they decided to modify their console version of the game by making the users pick one answer among multiple choices. Additionally, the developers added a pointing system and gave users the option to pick their songs as opposed to the TV show’s original structure where the songs are predetermined. To spice things up, the developers also added the Hint feature that took inspiration from another program called Who Wants To Be A Millionaire.

**Significance**

With Singing Bee, players can reminisce about the good old times by singing along the most iconic songs from the past decades up to the recent years. Everything about the game is carefully crafted to deliver a sentimental experience to users, from its extensive song selection to its nostalgic user interface design. Overall, Singing Bee aims to provide a fun for its users by creating a simple supplement-the-word singing-inspired game.

**Objectives**

The project aims to accomplish the following goals:

* To create a simple program that utilizes basic computer science concepts involving algorithms, logic, flowcharting, and coding
* To create a game that invokes a fulfilling joy in people of all generations and all walks of life
* To gain hands-on project experience in preparation for the workplace environment of the entire team

**Chapter 2**

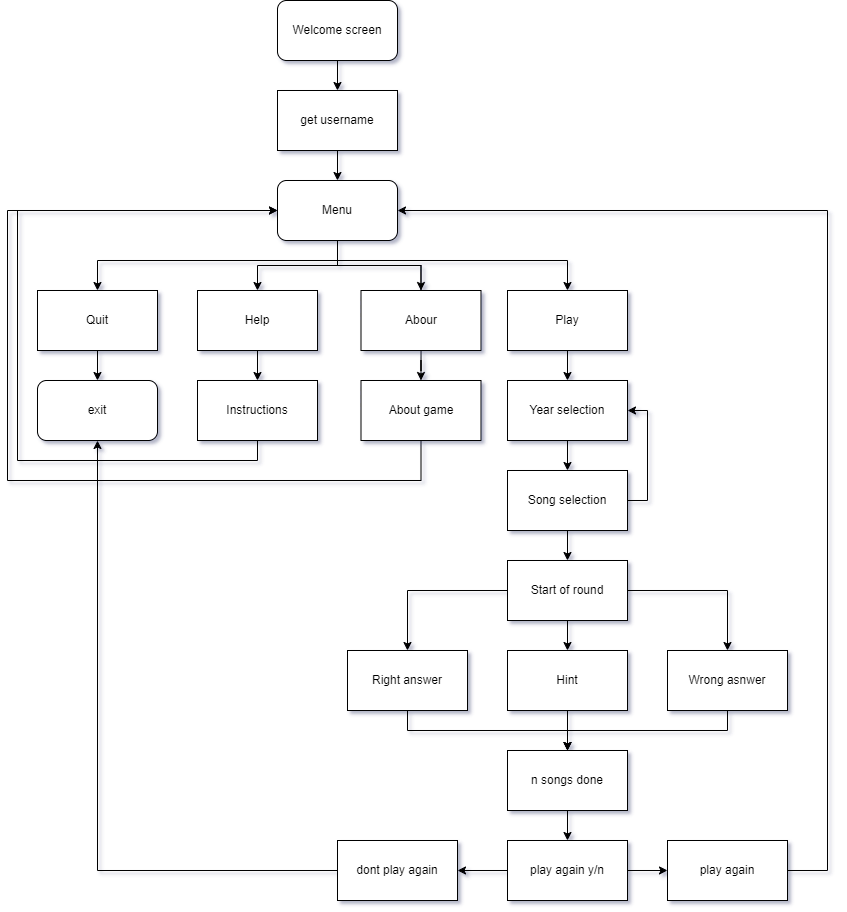
**METHODOLOGY**

**Program Design**

**Game Mechanics**

* + First, the player must enter a username to proceed to the Menu, which contains the following options: Play, About, Help, Quit. The game starts once the user chooses the “Play” option.
  + After selecting “Play”, the player moves to the Song Selection phase. The player adds tracks to their Song Cart by entering the correct keys. The songs come from a wide range of eras, from the 1960s to the 2020s. Each era has six representative songs.
  + Once the player is done curating their song line up and has confirmed it, they could now start playing the actual game by proceeding.
  + The chosen tracks play one-by-one in order. The lyrics also show up on screen. However, there are missing snippets that could be potentially filled by one of the four choices: A, B, C, D. The fifth choice is a Hint [H].
  + If the player gets a correct answer without using a hint, they will get 1000 points. However, If they get a correct answer with the help of a hint, they will only get 500 points. If the player’s answer is wrong, the player will get 0 points on that round.
  + After the n songs are done, the system will print out the total score of the user, and will ask if the user wants to quit or play again and score will reset to 0.

**Userflow**

****

**Workflow**

Below are steps the developers took to successfully build the Singing Bee project:

1. Brainstorming

The developers used the Figma Jamboard to bounce ideas off of each other in the earliest stage of project development. They eventually came up with 12 potential project ideas. Through a voting process, the team decided on the top two ideas to pitch to the adviser, Sir Mercado, for approval. The two projects were entitled “Animenatics” and “Singing Bee”.

1. Flowcharting

After presenting the project proposal to Sir Mercado and receiving his generous feedback, the team decided to push through with “Singing Bee” as the final project. To visualize the algorithm and logic needed for the program, the team organized a project flowchart.

1. Pseudocode

To complement the project flowchart, the team wrote pseudocode as a guide that aids in the coding process.

1. Development

All members of the team wrote code for various parts of the program such as the Game Menu, Song Selection, and Start Round phases. This portion of code is stored in the main.py file. They also added a JSON file entitled songList where song names are stored together with their singers and lyrics.

1. Testing

The Quality Assurance team actively ran and re-ran the code and ensured that the program was running smoothly. All bugs, errors, and logical discrepancies discovered were reported, documented, and resolved.

1. Revision

After receiving feedback from the Quality Assurance analysts, the developers edited the respective code blocks that needed attention.

1. Deployment

After preparing the documentation and successfully resolving major and minor issues within the program, the team launched the project on GitHub. Any user feedback will be taken into account and carefully considered for future development.

**Division of Labor**

In the creation of this project, we decided as a group to have a proper division of labor so as to help in making the project a lot more efficient and more organized as opposed to an unorganized workflow. In this section, we talk about the roles and the role pairings distributed among the members of the team.

**Roles**

The general work distribution is based on the following roles:

Project Management

* Project Management team’s duties include, but are not limited to, organizing the meeting schedules and due dates, working on the documentation of the project, and creating PowerPoint presentations.

Quality Assurance

* Quality Assurance team will make sure to test the program multiple times to search for bugs & errors, provide an update about the program

User Interface

* The User Interface team aims to improve the human-computer interaction and communication with the device and will make sure but not limited to improve the overall User experience.

**Pairings**

Apart from the above mentioned roles, every developer in the team also had to write code. In an attempt to divide the work equally, the team implemented the following task designations:

Game Menu

Espinoza, Arvin S.

Tomate, Alexene S.

Song Selection

Lumanog, Ian M.

Manalo, Irish M

Start Round

Francisco, Earl Ace Domand F.

Cunanan, Given V.

Mendoza, Jaosch Nathan C.

**Tools and Platforms**

1. Version control system
   1. Git/Github
2. Flowchart
   1. Figma
   2. Draw.io
3. Pseudocode
   1. Visual Studio Code
   2. Carbon.now
4. Communication
   1. Discord
   2. Google Meet
   3. Confluence

**Chapter 3**

**CONCLUSION**

**Timeline of Progress**

Overtime, the developers made the following progress:



Figure 1.1 Project Timeline for Singing Bee

Figure 1.1 shows the gantt chart of the progress made overtime with regards to the Singing Bee project. In the second week of November, the brainstorming phase commenced. It ended on the third week, and was followed by a short period of stagnation where the team waited for Sir Mercado’s feedback. After receiving his go-signal, the developers proceeded with flowcharting from Week 2 to Week 4 of December 2022. Pseudocode creation followed shortly after, starting from Week 4 of December 2022 to Week 2 of January 2023. The initial development phase had simultaneously occurred within the same time period.as Flowcharting and Pseudocode. Once the backbone of the program had been developed, the team started the Testing phase where the QA Analysts looked for bugs to resolve during the Revision phase. Both the Testing and Revision phases occurred from Week 1 to Week 2 of January. Lastly, the Deployment phase started and ended on Week 2 of the same month.

**Results**

After weeks of project-building, the developers produced the final outcomes as recorded in their GitHub repository, which can be accessed via the link below.

<https://tinyurl.com/SingingBeeRepo>



**Challenges and Resolutions**

Below are some difficulties and challenges encountered in the midst of project-building as well as the approaches the developers took in order to solve them:

**Program-wise**

1. Programming Language Used
   1. The programming language used became a challenge for the entire team when they had to use a completely separate programming language to the ones that they are more accustomed to and are more adept in. It then resulted in the misplacing of different programming syntaxes that are not present or plain wrong in the programming language used currently.
   2. In order to combat this, the entire team had to hone their skills in the usage of the programming language currently used in this project by continually working on the project so as to help lessen the amount of mistakes made in the creation of the project.
2. Bugs & Errors
   1. The user does not have any hints after playing again
   2. The player can proceed to the start of the round without choosing any songs
   3. The program has no total number of correct scores
   4. The user can add the same song, and it will be counted in the song cart

**Team-wise**

1. Miscommunication
   1. Miscommunication proved to be a very tenacious problem between the team as it has persisted in existing even after laying countermeasures in order to not experience any form of miscommunications between the team but with intentions being miscommunicated and ideas being misinterpreted, workflow efficiency took a hit as a result and overall project making was slowed down as a result.
   2. With the entire team noticing how destructive miscommunication is on their project making, they came to a conclusion that miscommunication between the entire team should be addressed and they agreed upon themselves that it is more optimal to be more clear, loud, direct, and less vague on any information, idea, suggestions and qualms that they have on the project or on the team.
2. Time Availability
   1. The individual time availability of each team member also became a problem that the team needed to surmount since workflow efficiency is stunted due to the members not having overlapping schedules to work on their parts due to other work and personal things that they need to do.
   2. In response to this, the team agreed on coming up with schedules that work with the members to work on their parts ahead of time and making sure that no other work overlaps with those set schedules.
3. Time Management Skills
   1. Another challenge encountered by some members of the team has to deal with their time management skills. Some members of the team have poor time management skills that affect their priorities when it comes to working on the project or when working on other things - mostly personal matters.
   2. Just like how they dealt with their time availability issues, they agreed on setting up schedules that prioritizes the project whilst still making time for other matters that the members are still destined to do.

**Recommendations**

Some of the areas that the developers can improve are:

**Program-Wise:**

* + Ideas for Future Development
    - Leaderboard - records player point and ranking history
    - Database - storage for song data (lyrics, artists, hints, etc.) as well as the user data (usernames, points, etc.)
    - APIs - contains special features that add more thrill to the program
      * Lyrical Construction
        + LyricsGenius
        + TheAudioDB

MusixMatch

* + - * Speech-to-Text
        + Rev.AI
        + AmberScript
        + Google Cloud’s Speech-to-Text
      * Thesaurus
        + WordsAPI
        + LinguaRobot
        + API Ninja - Thesaurus
    - Local Multiplayer Mode
    - Graphic User Interface (GUI)

**Team-Wise:**

* + Hard Skills
    - Solid Programming Foundations
    - Problem-solving Skills
    - Writing Clean Code
    - Technical Reading
    - Researching
  + Soft Skills
    - Clear Communication
    - Careful Listening
    - Proactiveness
    - Punctuality & Diligence