**Gateway to the wor♩d of music**

Music is an essential part of our lives in many forms, whether through singing, playing an instrument, or simply enjoying it in various settings or when we're in a particular mood. Having been passionate about this vast and beautiful field, I decided to create an application that introduces users to the world of music. It provides an opportunity to learn basic concepts and then apply them on a virtual piano, and there are also some songs that can be played and saved high scores.

**Project Flow**

Start (Form 1);

Authenticator (Form 7);

If (Teacher)  
 ManageClass (Form 8);

Else

{

Summary (Form 2);

If (Music Theory)

Theory (Form 5);

Else if (Try it out)

Piano (Form 3);

Else if (Test it)

{

Testing (Form 4);

Play (Form 6);

}

}

**Code Decisions & Features by Classes**

* AppDbContext
  + created the connection between the project and a database
* Student
  + Has a method that returns the smallest index available, in order to close gaps after deletion
* Teacher
* Has a method that returns the smallest index, i.e. the number of objects + 1
* Note
* Each note has an alteration (accidental) and an altitude
* The dictionary is used when converting the read songs from the file
* An absolute\_value method is necessary because a key can be written and saved in more than one way, even though it is always the same key. This problem appears only to the black keys, and this function would return for example 145, if it is the black key between 150 and 140. This method also takes into account the scale of the song, because if it has a key signature, the note will be saved without any accidentals, and it must not be mixed with the natural state of the note
* The draw\_it method draws the note on the staff in the given location, with everything that it needs
* Song
* Here is saved the list of all songs
* Staff
* Has methods for drawing the lines of the staff, the key signature, and also the current staff filled with notes
* Start
* Reads the songs from the file
* Authentication
* The username should be unique and every field must be at least 3 characters long
* Adds new students and teachers to the database
* Encrypts the password, which is not visible when typed
* ManageClass
* Only for the teacher, who can see the leaderboard and can delete students based on their username
* Summary
* The student can navigate between the 3 options: theory, practice and testing
* Tutorial
* Using the “Back” and “Next” buttons, a student can read about music theory while looking at suggestive images
* Piano
* The student can play up to 10 notes on the piano, being displayed on the staff. They can always be erased and giving the fact that every black key has 2 values, he can choose between the sharp previous note or the flat next note (for example C# and Db)
* Testing
* 10 random notes are generated, which should be played by the student. They can appear with accidents, and the method correctly\_generated ensures that every note is valid
* After playing some on the piano, the student can check them by pressing “Verify”, and his current score is displayed. If he manages to play all 10 accurately, he receives a congratulatory message
* Melody
* A random song is chosen, which should be played. At the end, its name is revealed.
* For each staff/line of the song, the student should play the same notes. In order to avoid confusion, if the absolute value of the note expected and the one played is the same, it will be drawn in the same form. Otherwise, the sharp version will be used.
* Playing a song correctly would bring significantly more points than trying to play the generated notes from the previous window, and additionally it is a more realistic exercise

**How to run the app**

Download it from <https://github.com/sorana5/GttWoM/tree/master> and put the “resources” file in bin/Debug/net8.0-windows (the main branch contains an older version of the app, running on .net framework 4.7, and the master branch has the .net 8 version).

**Improvements**

* I could have used more the name of the notes instead of the value (do or C, not 150)
* A user should be able to modify its data. A student should see the high score and the leaderboard. A student could have a certain teacher, so that there would be more classes. A student should also have a list of the songs that he played.
* Registration with an email address so that spam can be avoided, and it should be harder to create a teacher account. For the student account, there should be a code from the teacher such that he is in the right class.
* I could have used the relative position and size of the elements, so that the design would be suitable on more types of resolutions
* As future developments, apart from more songs and more theory (rhythm, scales, intervals, chords), I could start to work with durations, too, as now I was focus only on the right altitude of a note.

**Bibliography**

* https://csharp.agrimetsoft.com/exercises/Toggle\_Button\_Styling
* https://www.youtube.com/watch?v=V2QwIQxs0Mg
* https://www.youtube.com/watch?v=8pDBfDsqvQw
* https://www.musictheory.net/
* https://dev.to/hbolajraf/c-using-entity-framework-with-postgresql-database-41dg
* <https://www.youtube.com/watch?v=z7G6HV7WWz0>
* https://stackoverflow.com/