# Introduction

This document will provide general information about the test cases that group members generated. The test cases will ensure that musicXML application running successfully.

The application is a java-based program with graphical user-interface that will generate both machine-friendly MXL file, visual and sound information. The sound and visual information will allow user to understand machine-friendly XML file.

Due to the project is based on well-tested stater’s code, this document will force on two stages of additional functionality: staff generator and music player.

# Objective and tasks

The objective of the testing procedures is ensured that user will have error-free experience while they are using the software.

To reach is objective, testing procedures should finish following tasks.

1. Create test cases that simulate user input and valid output with desire output to ensure that program will handle user input correctly.
2. Create test cases that is possible for specific part of program that may crash the system to ensure that program will handle the errors correctly.

# Testing Strategy

## General approach introduction

Our testing is based on Junit framework and manually validation of result which is not machine friendly.

Our testing Strategy include two types of approach:

1. Overall test approach

This approach will consider the whole system as a “black box” that we only know the input and output. Test output will be generated atomically, but the output will be checked manually.

1. Unit test approach

This approach will teardown different part of system as an “unit”. We only consider possible input and output of specific unit; Test output will be generated atomically and valid by program automatically

Due to the limitation and cost of the program validation, unit test will not cover the methods that will generated human-friendly information (like visual and sound information)

## Overall test approach

### Player test

Testing method:

PlayerOutputTest.java will test the sound result generated with specific user input. A list of txt file that contain user input will be sent into system and running following test cases:

1. Generate music String
   1. Our player is based on JFugue library which required a music String to play. This test will generate music string automatically for group member to check manually
2. Generate sound (multi-Thread)
   1. The purpose of this test is to check if the player will crash the program. It will generate sounds in parallel to ensure that the testing won’t be limited by super long test instance.
3. Generate sound (single-Thread)
   1. The purpose of this test is to check if the player generates sound correctly .It will generate sounds one by one so that programmer can compare the sound generated by player with desired sound.

Each of above test case will be sent with 11 user input to ensure that it covers different user input.

### Visualizer test

Testing method:

visualizerOutputTest.java will test the visualized result generated with specific user input. A list of txt file that contain user input will be sent into system and running following test cases:

1. Generate pdf file
   1. The purpose of this test is check both file export function and visualized output. Each user input will be save as one “pdf” file into designed location for programmer to compare the outputs with desired results

Each of above test case will be sent with 11 user input to ensure that it covers different user input.

### GUI TEST

## Unit test approach

### Player test

### Visualizer test

1. ImageResourceHandler Test
   1. ImageResouceHandler is a

# Code coverage rate report

Our test cases cover over 89% methods in the visualizer and over 85% methods in the players, which is sufficient for programmer to identify problems in the program and provider solid user experience to the end-user.

