

## Test Plan and Report

- A. As a user, I want to be able to upload a file of music in order to eventually get sheet music back
- B. As a user, I want to have accurate sheet music so that I can play my music correctly
- C. As a user, I want to be able to save my generated sheet music so that I can revisit the sheets
- D. As a user, I expect a well-thought-out UI design using a designing tool such as Figma for the web application so that it is intuitive
- E. As a user, I want the web application to be visually appealing so that I have a good experience
- F. As a user, I expect the web application to have a functioning front-end so that I can have a good experience
- G. As a user, I expect the web application to be able to segment the audio file precisely so that it can identify notes
- H. As a user, I want the web application to be able to segment the audio files with other techniques so that it can classify more precise notes
- I. As a user, I expect the web application to be able to classify notes by giving it an audio file so that it can find the notes
- J. As a user, I want to be able to receive a pdf of generated sheet music so that I can reference it
- K. As a user, I want to be able to log into the web application
- L. As a user, I want to be able to save my sheet music on the web application so that I can reference the sheet music in the future
- M. As a user, I want the web application to have advanced technology such as ML technologies so that I can have the most precise sheet music
- N. As a user, I want the web application to be well-tested, so that I get the most precise sheet music
- O. As a user, I want the web application to well connected so that it is functioning
- P. As a user, I expect the web application to the well documented so that in the future if the application needs support their is documentation

### Scenario 1: Upload .wav file (Pass)

1. Start web application with npm start
  - a. file=d2.wav
2. User can click on upload button
3. File directory for their computer should pop up
4. They are able to select file
5. They are able to see successful upload pop up

### Scenario 2: Get accurate sheet music (Pass)

1. User can click on upload button
2. File directory for their computer should pop up
3. They are able to select file
  - a. file=d2.wav
4. They are able to see successful upload pop up
5. Wav file gets sent to backend, and gets converted into sheet music
6. User can click on the download button once converted and in the sheet music they can see if it's a d4

Scenario 3: Save generated sheet music (Pass)

1. User can sign up or login into their account from clicking on the button with the pop-up
2. Once logged into the account, user can click on upload button
3. File directory for their computer should pop up
4. They are able to select file
5. They are able to see successful upload pop up
6. Wav file gets sent to backend, and gets converted into sheet music
7. User can click on the download button once converted
8. There will be an option to save it for the account
9. On the webpage a button to see saved audio files which could be convert into sheet music again ....

Scenario 4: Well Designed *Subjective* (Pass)

1. Icons for uploading, downloading and logging in are intuitive

Scenario 5: Visually Appealing *Subjective* (Pass)

1. Icons for each of the features are visually appealing and intuitive to use

Scenario 6: Functional Frontend (Pass)

1. Buttons for Logging in, uploading and downloading are functional

Scenario 7: (Fail)

1. User can upload a file with complex rhythm
2. After the backend has finished processing the audio file the users should be able to click download pdf
3. The file downloaded would be sheet music with complex rhythm

Scenario 8: Complex Rhythm (Fail)

1. User uploads wav file of song with complex rhythm
2. User downloads pdf
3. Sheet music displays complex rhythm (different lengths of notes)

Scenario 9: (Pass)

1. User can click on upload button
2. They are able to select file

3. They are able to see successful upload
4. Wav file gets sent to backend, and gets converted into sheet music
5. User can click on the download button once converted and in the sheet music they can see if it's a d4

Scenario 10: Download .pdf file (Pass)

1. User can upload file
2. The frontend should display a processing audio icon
3. After the backend has finished processing the audio file the users should be able to click download pdf
4. The file should then appear in the user's download folder

Scenario 11: Login (Pass)

1. Select Login  
username="[molly@example.com](mailto:molly@example.com)"  
password="molly"  
Press Enter key
2. User should be directed back to home page and now be logged in

Scenario 12: Save sheet music (Fail)

1. User logs in
2. User uploads sheet music
3. Sheet music is saved to users account in database

Scenario 13: Machine learning accuracy (Pass)

1. User uploads low quality wav file
  - a. file = mic.wav
2. Sheet music is accurate

Scenario 14: Well tested (Pass)

1. Features of the web application have been tested

Scenario 15: Fully Connected Speed (Pass)

1. User uploads wav file
2. User downloads wav file
3. Frontend talking to backend does not take super long

Scenario 16: Documentation (Pass)

1. PDF files exist on the repository