MODEULE

AN ONLINE MUSICAL TOOL

BY CALLUM COOPER LEE

Table of Contents:

ANALYSIS 3

Description of Problem 4

AH Concepts 4

Scope 5

Boundaries 5

Constraints 6

Requirements Specification 7

Functional Requirements 7

End-User Requirements 9

UML Diagram 10

Project Plan 11

Timeline 11

Resources 11

Detailed plan 12

DESIGN 13

Design of User Interface 14

Media queries 17

Design of Background Processes 20

Pseudocode 42

Design of Integration 48

IMPLEMENTATION 52

FULLY IMPLEMENTED WEBSITE 53

FINAL PRODUCT (WIDTH > 720PX) 54

FINAL PRODUCT (WIDTH < 720PX) 67

ALL FILES 81

HTML CODE 83

CSS CODE 88

JAVASCRIPT CODE 93

PHP CODE 102

LOG OF ON-GOING TESTING 110

DEVELOPMENT OF NEW SKILLS AND KNOWLEDGE 112

TESTING 113

FINAL TEST PLAN 114

FUNCTIONAL REQUIREMENTS TESTING 123

USER REQUIREMENTS TESTING 150

USABILITY TESTING 162

EVALUATION 168

FITNESS FOR PURPOSE 168

MAINTAINABILITY 168

EVIDENCE OF MAINTAINABILITY 169

ROBUSTNESS 170

CONCLUDING THOUGHTS 170

Analysis

To analyse my website, I will begin by figuring out what I want to create. I will then begin to build on my idea and come to a detailed description of the problem. I will then decide what Advanced Higher Concepts to include in my project. It will be important to outline the scope, boundaries, and constraints of the project to clarify what will be included. I will define clear functional and end-user requirements. A UML will be created to show the interactions between users and the back end of my project. I will then create a full project plan to help me break down how to go about creating my project.

PREFIXES IN THIS SECTION	**DESCRIPTION**
F	Define a Functional Requirement
U	Define an End-User requirement

Description of Problem

I have decided to create a website for musicians to select and listen to different musical scales. My website will be called 'Modeule'. This is a mix of the words 'mode' (a type of musical scale), and 'module' (a word familiar to those in the field of computing science).
The interface will be large piano keyboard that a user can play. There will be a form from which a user can select a starting note, and a scale type. Once the user clicks a play button, the website will audibly play the notes in the correct order as the keys are pressed on the keyboard. I intend to include a register and login from and a button to favourite a scale (to give easier access to scales). The end user for my site would be a musician who wanted to familiarise themselves with the patterns and sounds of specific musical scales.

AH Concepts

My site will be Advanced Higher standard.

My project will major in website design and development (WDD). My website will be externally styled by CSS. My website will have multiple layouts using a media query to make the website functional with a different aspect ratio. I will include form elements using name and submit the form data to PHP using a method and action. In PHP I will use session variables to retain data until a user logs out.

My project will minor in database design and development (DDD). I will connect to a database in PHP. I will create two linked tables with primary keys with SQL. My code will manipulate the data within these tables using INSERT and DELETE queries. This will allow for users to enter data into a form to be stored in the database. I will also use SQL to retrieve data from a table using a SELECT query and format the results to determine an output.

Scope

- A full analysis of the problem. A UML diagram must be included to clarify the functionality of the site. A project plan will be maintained.
- A full design of the website, including wireframe designs and pseudocode for form processing. Wireframes will be made for multiple layouts, depending on the width of the browser.
- A fully implemented website with functioning server-side processing and databases to store audio files and user data.
- A sensible test plan including full system testing, usability, and integration testing. Usability testing will be completed using personas.
- A full comparison of predicted test results and final test results to ensure the final website matches the design.
- An evaluation of the final site, referring to usability, maintainability, and fitness for purpose.

Boundaries

- A user will have to log in or sign up to use the favourite functionality of the website.
- A login password must be between 8 and 20 characters and must contain at least one alphabet letter, one number and a special character such as $, % or *.
- The website will only be functional on several PC browsers.
- If the user holds down a key on the interface, the note played back will not loop.
- Only two octaves of notes will be available to the user.
- The website will be started by running Apache and MySQL on Xampp, then typing localhost/AH Project into the URL of a browser, then clicking on mainPage.php.
- The website will be tested in both Chrome an Microsoft Edge.

Constraints

Time: the entire project MUST be completed before and submitted to the SQA by the 1st April 2022.

Technical: I will be using XAMMP to host databases, and the Atom text editor to write my HTML, CSS, JavaScript and PHP code. I am not technically constrained, but I will need to expand my website development skills to complete the project. I will have access to my own laptop, the school computers, and my mobile phone.

Economic: All the software I will be using will not cost anything as I have legal prior access. I will work on this project for free, so my time on this project will not affect the cost. I will use Office 365 for the documentation of this project, as licenced by the school.

Legal: I will have to carefully pick images for my website to make sure I do not breach the copyright law. Another solution to this would be to take or create my own images. Licencing is not an issue for the required sound files, as I will record these sounds myself. The form will take in some personal details, which will be encrypted and not displayed in the URL for privacy. All software is licenced.

Requirements Specification

The purpose of my website will be to allow users to visualise and listen to a range of musical scales, starting on any note. The user will also be able to log in or register to 'favourite' specific scales. Favouriting a scale will make it easier for a user to retrieve specific scales quickly.

Functional Requirements

F1.	When a piano key is clicked: i. The corresponding note must be audibly played ii. The key must appear to be pressed, imitating a real piano iii. The name of the note must be displayed as text
F2.	When the scale form is complete and play button clicked: i. The starting note and the scale type must be collected from the form ii. The scale must be played up and down an octave iii. The scale being played will be displayed as text
F3.	When a user is not logged in: i. The favourite inputs and the logout button will be disabled ii. The user will be able to click a 'Login' or 'Register' button iii. The keyboard, scale form and play button will be functional
F4.	When the 'Login' button is clicked: i. A username, password and submit input will become available to the user ii. A value will be stored to tell the PHP to login, not to register
F5.	When 'Login' form is filled in and submitted: i. The username and password will be submitted to PHP ii. Get correct password from the 'users' table iii. If both username and password are valid, log in the user iv. If username or password is invalid, return to start page
F6.	When the 'Register' button is clicked: i. A username, password and submit input will become available to the user ii. Password assistance will appear to ensure security of the new password iii. A value will be stored to tell the PHP to register, not to login
F7.	When 'Register' form is filled in and submitted: i. The username and password will be submitted to PHP ii. If password is secure, add new user to the 'users' table iii. Login the new user

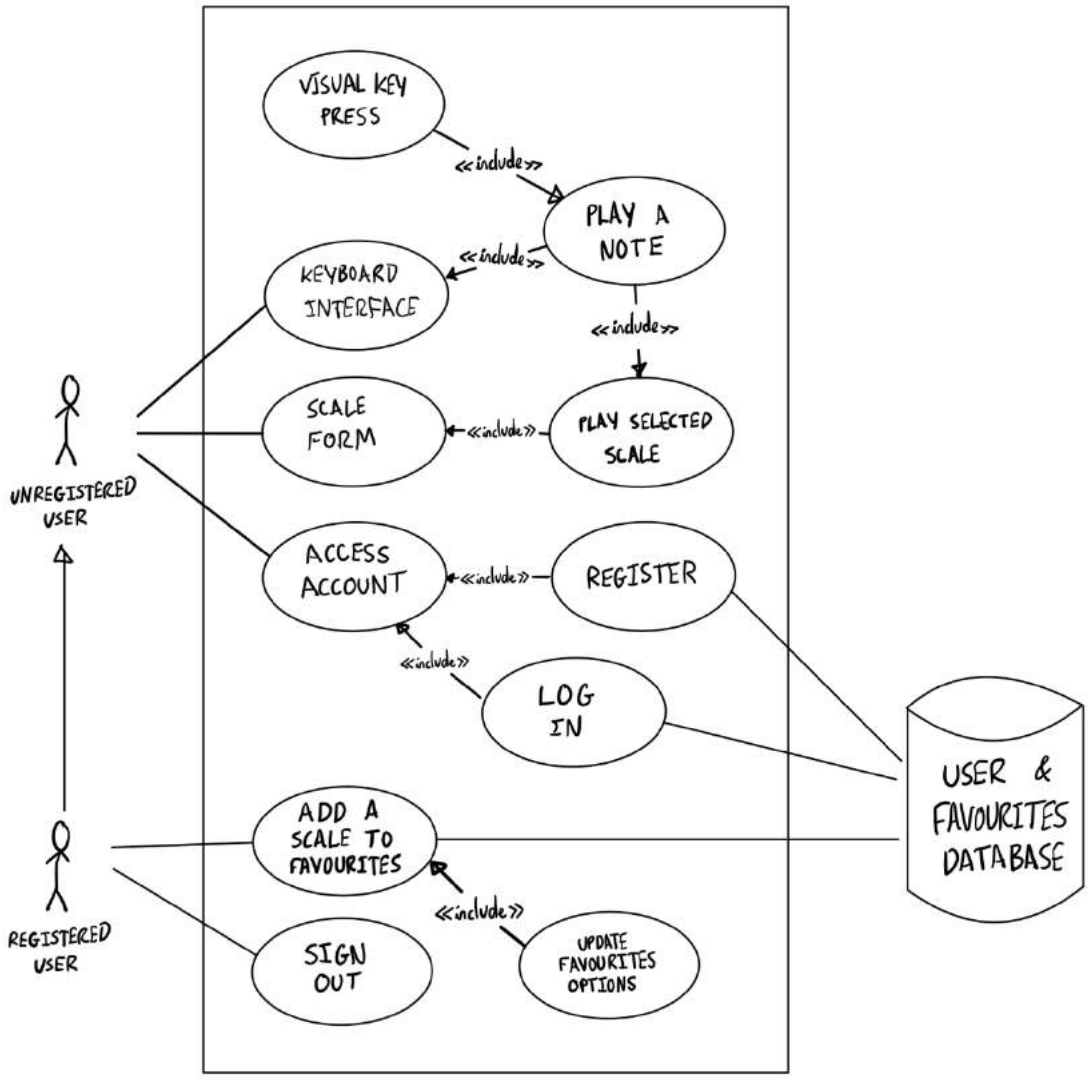
F8.	When a user is logged in: i. The favourite inputs and the logout button will be functional ii. The 'Login' and 'Register' buttons will be disabled iii. The keyboard, scale form and play button will be functional
F9.	When the scale form is complete and favourite button clicked: i. The starting note and the scale type must be collected from the form ii. If the scale entered already exists in the 'favourites' table, delete it iii. If the scale entered is new to the 'favourites' table, add the new data
F10.	When an option is selected from the 'favourites' list: i. Collect scale data from the selected option ii. Fill out the scale form with the scale data
F11.	The favourite button's appearance must be up to date with the scale form.
F12.	When the 'Logout' button is pressed, log the user out and return to the base page

End-User Requirements

To identify the end-user requirements, I used user scenarios. As a musician, it was easy for me to identify what a user might want and need from this website:

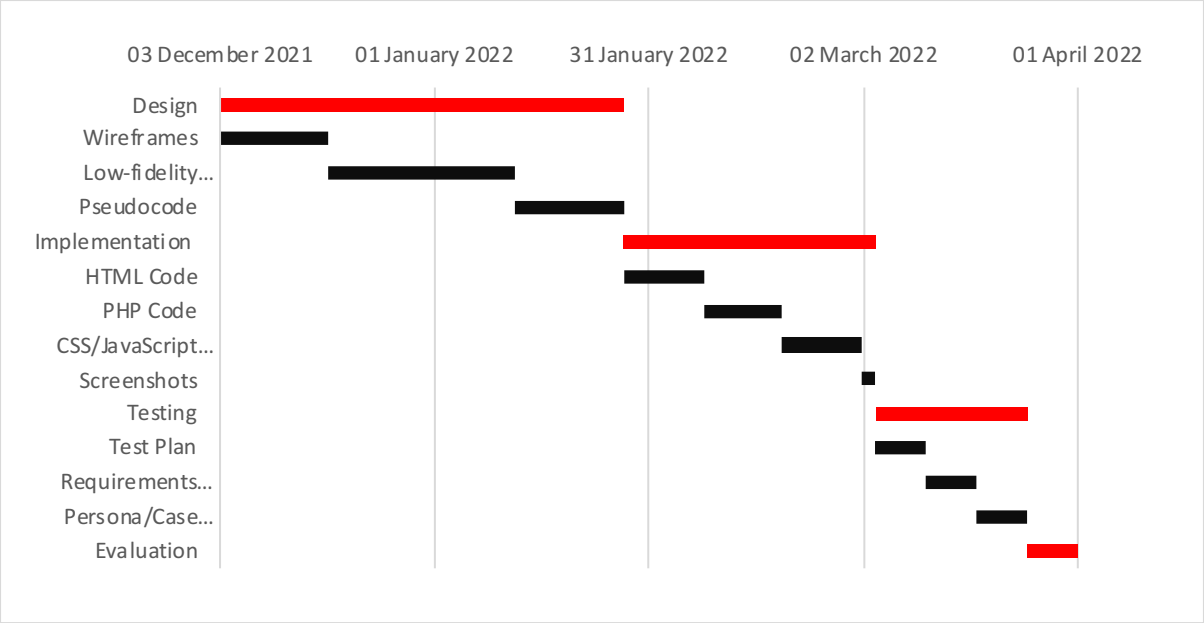
U1.	The user will be able to interact with a keyboard interface and play individual notes by clicking on the corresponding key. The user will be able to see and hear a note being played.
U2.	The user will be able to choose a scale to play by filling in a form.
U3.	After finishing the form, the user will be able to press a 'play' button. The user will then hear the selected scale played back. Each note of the scale should be heard by the user and simultaneously the same key should look as if it has been pressed.
U4.	The user will be able to see the name of the note being played and the scale that is selected as text.
U5.	The user will be able to access a log in form or register form by clicking on a suitably labelled button.
U6.	A user will be able to fill in the register form. As they fill in the password, they should be able to visually see if their password is valid.
U7.	A user will be able to submit the register form. If the password is valid, they should be added to the 'users' database and logged in. There should be no output if the password is invalid.
U8.	A registered user will be able to fill in and submit the login form. They will be logged in and gain access to the favourites feature of my website.
U9.	Logged-in users will be able to click a heart icon to favourite a scale.
U10.	A logged-in user will be able to click on the favourite button when the scale is already favourited to remove it from their favourites.
U11.	A user will easily be able to identify if the current scale is favourited. If a scale is already favourited, the heart icon will be filled. If the scale is not favourited, the icon will be hollow
U12.	Logged-in users will be able to choose a favourited scale from a dropdown list. When an option is selected, it will automatically fill in the scale form with the scale.
U13.	Logged-in users will be able to log out.

UML Diagram


VISUAL KEY PRESS
«include»
PLAY A NOTE
«include»
KEYBOARD INTERFACE
«include»
SCALE FORM
«include»
PLAY SELECTED SCALE
UNREGISTERED USER
ACCESS ACCOUNT
«include»
REGISTER
«include»
LOG IN
USER & FAVOURITES DATABASE
ADD A SCALE TO FAVOURITES
«include»
UPDATE FAVOURITES OPTIONS
REGISTERED USER
SIGN OUT

Project Plan

Timeline



Resources

All stages require access to Microsoft Word software for documentation.

Analysis:

- Access to graphing / design tools

Design:

- Access to wireframe design tools

Implementation:

- Access to computer with programming software
- Access to a browser for testing

Testing:

- Access to programming software to correct errors and monitor code
- Requires non-biased people for user testing.

Detailed plan

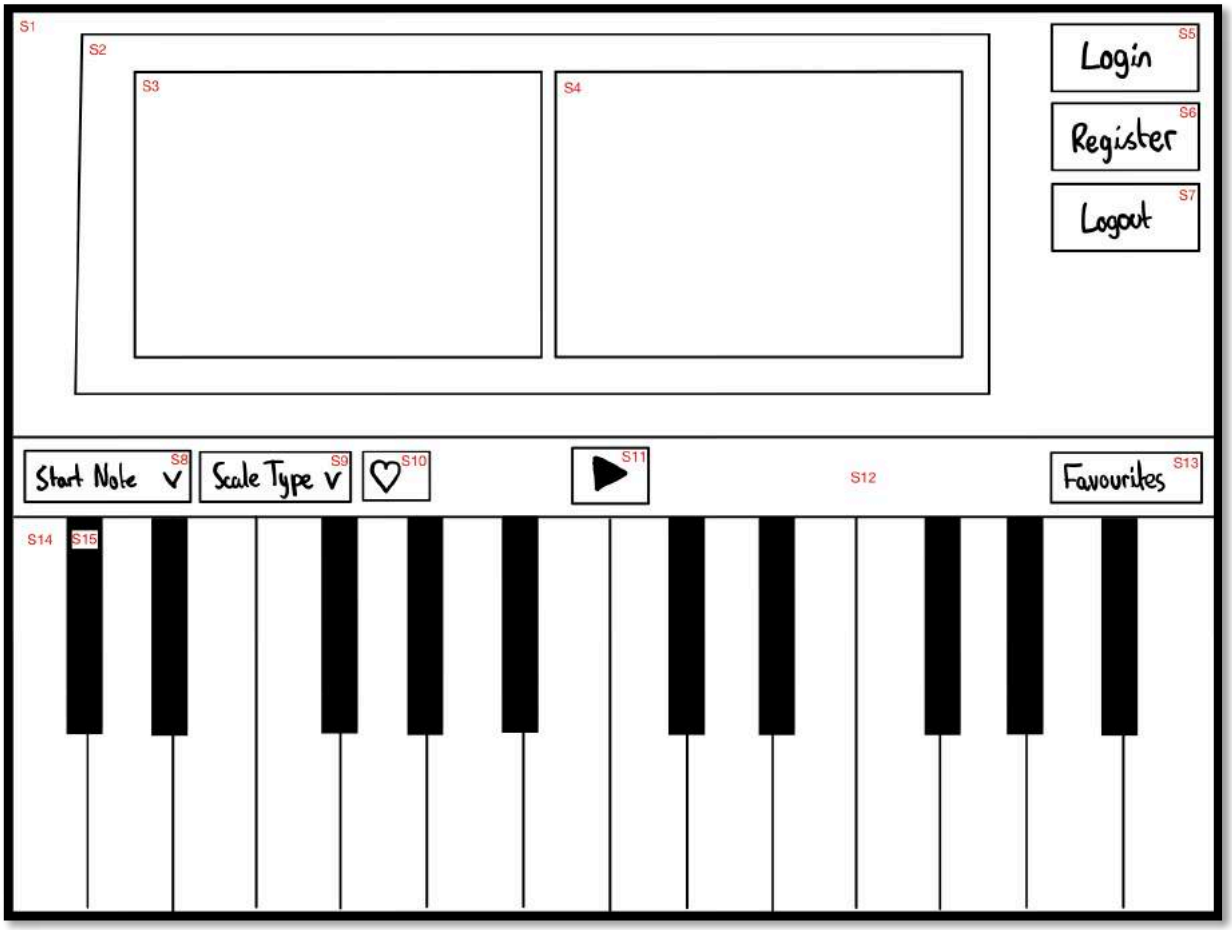
DESIGN	Design wireframes
	Design media queries
	Design background processes
	Design integration
INTEGRATION	Create base website
	Add Piano
	Create databases
	Create login / register service
	Create logout button
	Create favourite functionality
TESTING	Create final test plan
	Requirements Testing
	Usability Testing
EVALUTION	Fitness for Purpose
	Maintainability
	Robustness
	Concluding Thoughts

Design

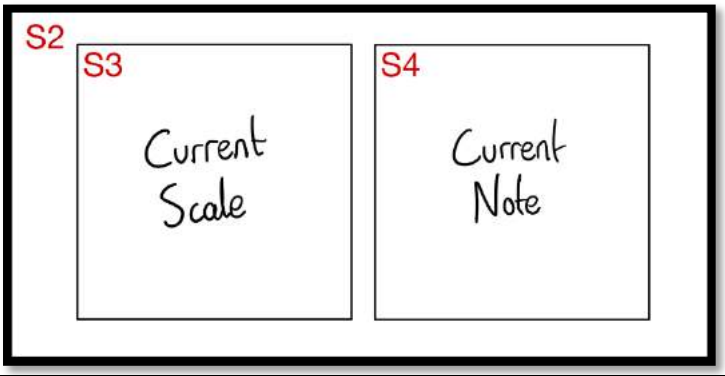
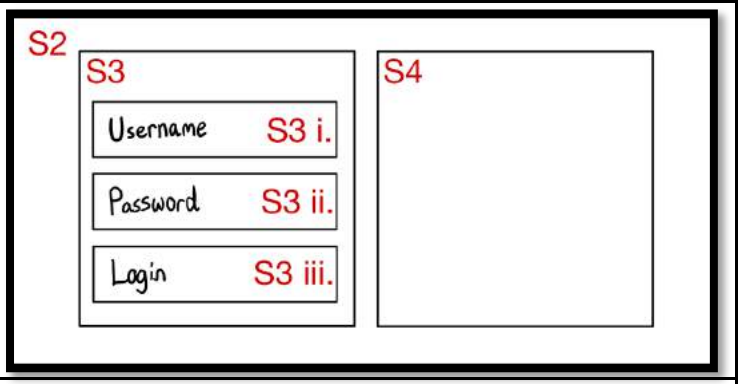
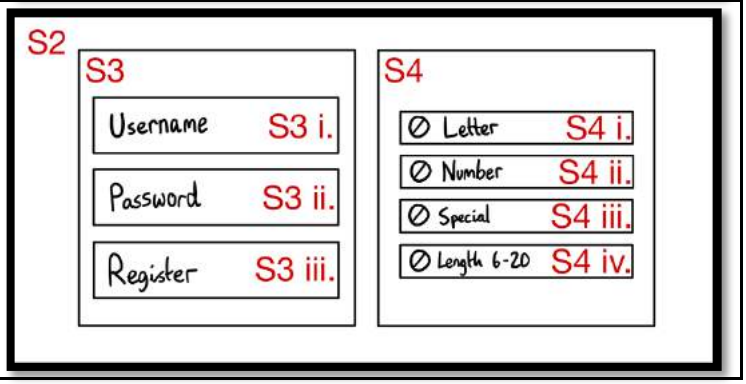
To design my website, I will begin by designing the user interface. I will create a wireframe and label elements with the prefix S. This will help me reference changes to styles and functionality of elements. I will then design my media queries, providing a labelled wireframe to demonstrate how the website will look with a narrower window. The next step will be to design the background processes. I will identify the requirements that an element will be meeting. All available inputs to the user will be identified and will trigger an output. Outputs will be identified with the prefix P and will be written out in pseudocode. I will design queries to create tables in my database. I will also design queries to interact with the database, identified by the prefix Q. These queries can be used in the pseudocode to integrate with the database.

PREFIXES IN THIS SECTION	**DESCRIPTION**
F	Functional Requirement has been met
U	End-User requirement has been met
S	Used to identify a singular element
P	Used to identify a process that has been written in pseudocode.
Q	Used to identify a query to integrate with the database

Design of User Interface



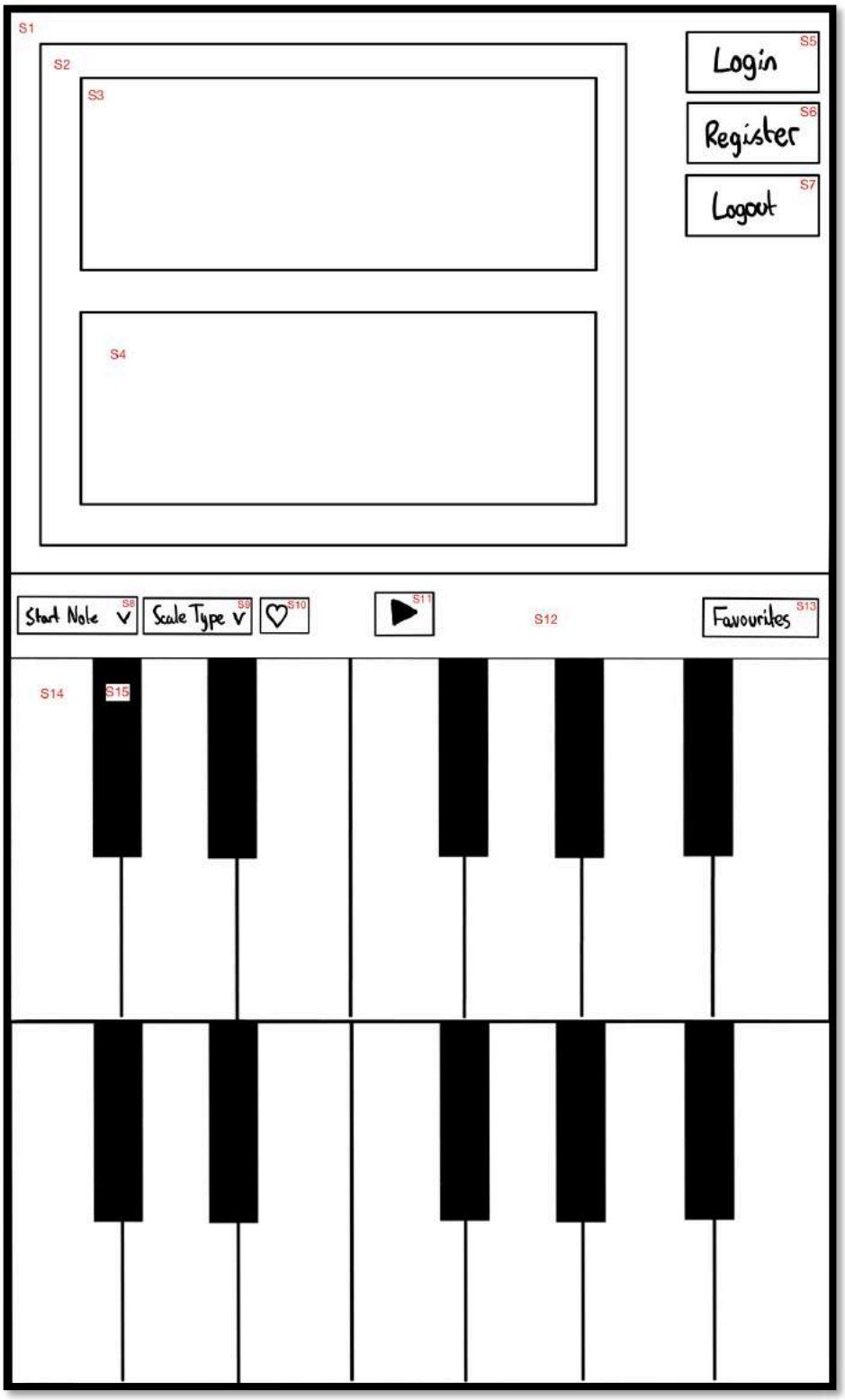
My website will only have one page. To accomplish this whilst still meeting all the requirements, the element S2 has three different layouts. It is a skeuomorphic representation of a digital screen that you would find on some keyboards. By default, it will display the selected scale and the last note played. If the user intends to log in or register, the forms to do so will be displayed on the 'screen' instead of the default layout.

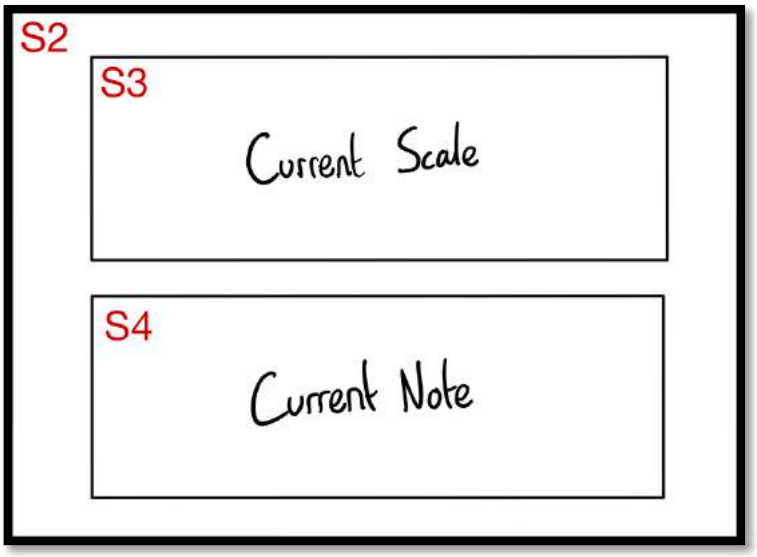
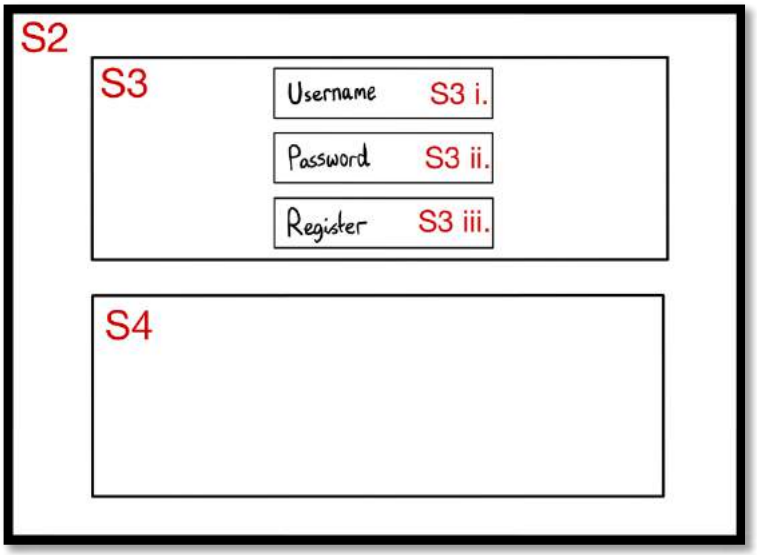
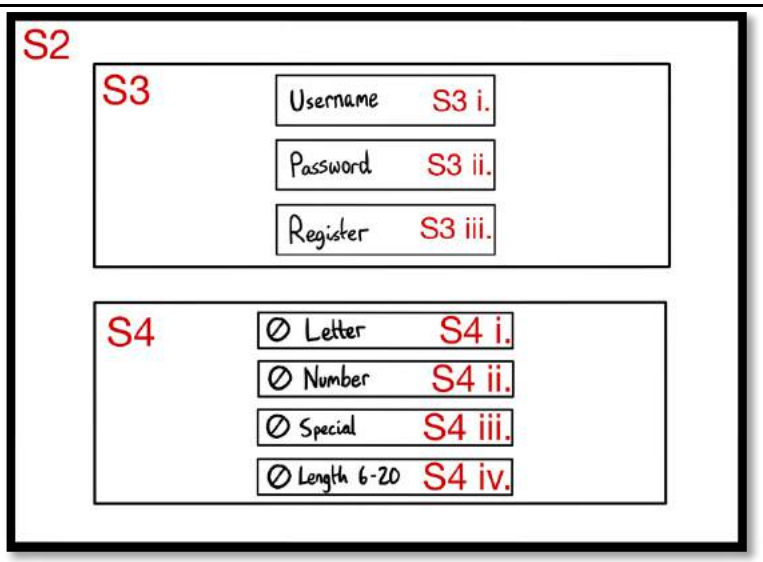
S2 Layout	Description	Wireframe
A	The default layout. It will revert to this whenever a key is played, the scale form is filled out or when a user logs out. Logged-in users will only see this layout.	
B	The Login form. This layout is used when a user clicks on the 'Login' button. S4 is empty for this layout.	
C	The Register form. This layout is used when a user clicks on the 'Register' button. S4 will inform the user if the password they are entering is valid.	

CSS STYLING		
Element Label	**Element Description**	**CSS**
S1	Overall background with a wood like texture.	background-image: //FILE NAME// background-size: 100% 100%; background-repeat: no-repeat; color: red;
S2	Screen-like display with reddish text. It is intended to look like a screen on an old synth.	background-color:#2b0e10; width:auto; height:40%;
S3 S4	Splits the screen in two, allowing to display all key information. Changes when "Login" or "Register" are clicked.	Width: 50%; Height: 100%;
S5 – S11 S13	All inputs that are not on the screen interface. Styled to look like they are lit / unlit depending on if they are active or disabled respectively.	height:20px; font-size:125%; Background-color: #ffaa33; When disabled: Background-color:#452c0a
S5 – S9 S13	Select and Submit buttons that are not on the screen interface.	Width: 10%;
S10 S11	Play and pause buttons	width:30px;
S12	The bar containing the scale form and the favourite buttons	background-image: //FILE NAME// height:10%; bottom:47%;
S14 S15	Black and white key sizes, All determined by media queries	width: keyWidth; height: keyHeight;
S14	White Key	Background-color: white;
S15	Black Key	Background-color: black;

Media queries

When the width of the browser is less than 720px:



S2 Layout	Description	Wireframe
A	The default layout. It will revert to this whenever a key is played, the scale form is filled out or when a user logs out. Logged-in users will only see this layout.	
B	The Login form. This layout is used when a user clicks on the 'Login' button. S4 is empty for this layout.	
C	The Register form. This layout is used when a user clicks on the 'Register' button. S4 will inform the user if the password they are entering is valid.	

CSS STYLING	**@media only screen and (max-width:720px)**	
Element Label	**Element Description**	**Element Style**
S1	Overall background with a wood like texture.	background-image: //FILE NAME// background-size: 100% 100%; background-repeat: no-repeat; color: red;
S2	Screen-like display with reddish text. It is intended to look like a screen on an old synth.	background-color:#2b0e10; width:auto; height:40%; font-family: "Press Start 2P", sans-serif;
S3	Splits the screen in two, allowing to display all key information. Changes when "Login" or "Register" are clicked.	Width: 50%; Height: 100%;
S4, S5	All inputs that are not on the screen interface. Styled to look like they are lit / unlit depending on if they are active or disabled respectively.	height:20px; font-size:125%; Background-color: #ffaa33; When disabled: Background-color:#452c0a
S4	Select and Submit buttons that are not on the screen interface.	Width: 10%;
S5	Play and pause buttons	width:30px;
S6	The bar containing the scale form and the favourite buttons	background-image: //FILE NAME// height:10%; bottom:47%;
S7 S8	Black and white key sizes, All determined by media queries	width: keyWidth; height: keyHeight;
S7	White Key	Background-color: white;
S8	Black Key	Background-color: black;

Design of Background Processes

S1		
DESCRIPTION OF ELEMENT		
Overall background with a wood like texture.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUTS
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
N/A		

S2		
DESCRIPTION OF ELEMENT		
Screen-like display with reddish text. It is intended to look like a screen on an old synth. It is used to display which scale is being played and which note has been played as text. It will also contain the Login/Register forms when a user clicks 'Login' or 'Register'.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F1 iii. F2 iii. F4 i. F6 i. , ii.	U4 U5 U6 U7 U8	

S3		
DESCRIPTION OF ELEMENT		
The first half of the element S2. Displays either the 'Current Scale' as text or the Login/Register form.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F2 iii. F4 i. F6 i.	U4 U5 U6 U7 U8	

S3 i.		
DESCRIPTION OF ELEMENT		
Username input		
UNDERLYING PROCESSES		
INPUTS	**PROCESSES**	**OUTPUT**
Name = username Type = text	N/A	N/A
VALIDATION		
Required Minlength = 1 Maxlendth = 20		
LINKED REQUIREMENTS		
FUNCTIONAL	**END-USER**	
F4 i. F5 F6 i. F7	U5 U6 U7 U8	

S3 ii.
DESCRIPTION OF ELEMENT
Password input
UNDERLYING PROCESSES

INPUTS	PROCESSES	OUTPUT
Name = password Type = password	Output when a character is entered into the field.	P1

VALIDATION
Required Minlength = 6 Maxlendth = 20 Contains a letter, number and special character
LINKED REQUIREMENTS

FUNCTIONAL	END-USER
F4 i. F5 F6 i. F7	U5 U6 U7 U8

S3 iii.
DESCRIPTION OF ELEMENT
Submit button for Login / Register form. The value of this button will determine whether the username and password are used to log in or to register a new user.
UNDERLYING PROCESSES

INPUTS	PROCESSES	OUTPUT
Name = enterDetails Type = 'Submit'	Output when clicked	P2

VALIDATION
N/A
LINKED REQUIREMENTS

FUNCTIONAL	END-USER
F4 i. F5 F6 i. F7	U5 U6 U7 U8

S4		
DESCRIPTION OF ELEMENT		
The second half of the element S2. Displays either the 'Current Note' as text or the password validation feedback when a user is registering.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F1 iii. F6 ii.	U4 U6	

S4 i.		
DESCRIPTION OF ELEMENT		
The element to indicate if the password contains a letter or not		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F6 ii.	U6	

S4 ii.		
DESCRIPTION OF ELEMENT		
The element to indicate if the password contains a number or not		
UNDERLYING PROCESSES		
INPUTS	**PROCESSES**	**OUTPUT**
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	**END-USER**	
F6 ii.	U6	

S4 iii.		
DESCRIPTION OF ELEMENT		
The element to indicate if the password contains a special character or not		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F6 ii.	U6	

S4 iv.		
DESCRIPTION OF ELEMENT		
The element to indicate if the password is between 6 and 20 characters long		
UNDERLYING PROCESSES		
INPUTS	**PROCESSES**	**OUTPUT**
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	**END-USER**	
F6 ii.	U6	

S5		
DESCRIPTION OF ELEMENT		
Button to bring up the login form.		
UNDERLYING PROCESSES		
INPUTS	**PROCESSES**	**OUTPUT**
Name = loginType Value = Login Type = Submit	Output when clicked.	P5
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	**END-USER**	
F3 ii. F4	U5	

S6
DESCRIPTION OF ELEMENT
Button to bring up the register form.
UNDERLYING PROCESSES

INPUTS	PROCESSES	OUTPUT
Name = loginType Value = Register Type = Submit	Output when clicked.	P5

VALIDATION
N/A
LINKED REQUIREMENTS

FUNCTIONAL	END-USER
F3 ii. F6	U5

S7		
DESCRIPTION OF ELEMENT		
Button to Logout		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = logout Type = Submit	Output when clicked	P6
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F12	U13	

S8		
DESCRIPTION OF ELEMENT		
Select input to choose the note that the scale will begin on.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = startNote Type = Select	Output when any option is selected by the user.	P7
VALIDATION		
Required		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F2 i. F9 i. F11	U2 U4 U9 U11 U12	

S9		
DESCRIPTION OF ELEMENT		
Select input to choose the type of scale that will be played.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = scaleType Type = Select	Output when any option is selected by the user.	P7
VALIDATION		
Required		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F2 i. F9 i. F11	U2 U4 U9 U11 U12	

S10		
DESCRIPTION OF ELEMENT		
The 'favourite' or like button. When the scale form is filled in and this button is pressed, the scale will be favourited. If a scale is already favourited, the scale will be removed from the user's favourites.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = favourite Type = submit	Output when clicked.	P8
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F9	U2 U4 U9 U12	

S11		
DESCRIPTION OF ELEMENT		
The 'Play' button. When the scale form is filled in and this button is pressed, the scale will be played back to the user.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = scale Type = submit	Output when clicked.	P9
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F2	U3	

S12		
DESCRIPTION OF ELEMENT		
The bar above the keyboard. This contains the scale form and the favourite inputs.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	N/A	N/A
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
N/A	N/A	

S13		
DESCRIPTION OF ELEMENT		
Select input for a user to choose one of their favourites.		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
Name = favouritesList Type = Select	Output when any option is selected by the user.	P10
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F10	U12	

S14		
DESCRIPTION OF ELEMENT		
A white key		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	Output when clicked.	P11
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F1 F2 ii.	U1 U3	

S15		
DESCRIPTION OF ELEMENT		
A black key		
UNDERLYING PROCESSES		
INPUTS	PROCESSES	OUTPUT
N/A	Output when clicked.	P11
VALIDATION		
N/A		
LINKED REQUIREMENTS		
FUNCTIONAL	END-USER	
F1 F2 ii.	U1 U3	

Pseudocode

INDEX	PSEUDOCODE
P0 (init)	1. Initialise screen layout 1.1. Set element S2 to Layout 3 2. Set all forms to default 2.1. Enable all form elements 2.2. Disable element (S7) 2.3. Disable element (S10) 2.4. Disable element (S13) 3. If the user is logged in, enable favourite forms and logout button 3.1. IF session['loggedOn'] == true THEN 3.1.1. Enable element (S7) 3.1.2. Enable element (S10) 3.1.3. Enable element (S13) 3.1.4. Disable element (S5) 3.1.5. Disable element (S6) 4. Calculate and set the sizes for all the keys 4.1. SET screenWidth TO width of screen in px 4.2. IF screenWidth > 720px THEN 4.2.1. SET keyWidth TO screenWidth / 24 4.2.2. SET keyHeight TO 50% 4.2.3. Create 24 piano keys 4.3. ELSE 4.3.1. SET keyWidth TO screenWidth / 12 4.3.2. SET keyHeight TO 25% 4.3.3. Create 2 stacked sets of 12 piano keys 4.4. END IF

P1	**1.** Collect the value of the password **2.** SET session['valid'] TO false **3.** Check if the password is valid **3.1.** IF the password contains a letter AND a number AND a special character AND is 6-20 characters long THEN **3.1.1.** SET session['valid'] TO true **3.2.** END IF **4.** Update the password validity indicators in the element S4 **4.1.** IF the password contains a letter THEN **4.1.1.** Update S4 i. to have a check mark **4.2.** ELSE **4.2.1.** Update S4 i. to have a cross icon **4.3.** END IF **4.4.** IF the password contains a number THEN **4.4.1.** Update S4 ii. to have a check mark **4.5.** ELSE **4.5.1.** Update S4 ii. to have a cross icon **4.6.** END IF **4.7.** IF the password contains a special character THEN **4.7.1.** Update S4 iii. to have a check mark **4.8.** ELSE **4.8.1.** Update S4 iii. to have a cross icon **4.9.** END IF **4.10.** IF the password is 6 – 20 characters long THEN **4.10.1.** Update S4 iii. to have a check mark **4.11.** ELSE **4.11.1.** Update S4 iv. to have a cross icon **4.12.** END IF

P2	1. Get the username and password and set as session variables 1.1. SET session['username'] TO post['username'] 1.2. SET session['password] TO post['password'] 2. Check if the user is to be logged in or registered 2.1. IF post['enterDetails'] == 'Login' THEN 2.1.1. Perform output **P3** 2.2. END IF 2.3. IF post['enterDetails'] == 'Register' THEN 2.3.1. Perform output **P4** 2.4. END IF
P3	1. If the username exists in the table, collect the password from the same record 1.1. SET password TO Query **Q1** 1.2. IF password == null THEN 1.2.1. DISPLAY ERROR 1.2.2. STOP PROGRAM 1.3. END IF 2. Validate the password and set an appropriate session variable 2.1. IF password == session['password'] THEN 2.1.1. SET session['loggedOn'] TO true 2.2. ELSE 2.2.1. SET session['loggedOn'] TO false 2.3. END IF
P4	1. If the password is valid, add the new user to the table 1.1. IF session['valid'] == true THEN 1.1.1. Query **Q2** 1.2. END IF 2. Log the new user in 2.1. Perform output **P3**

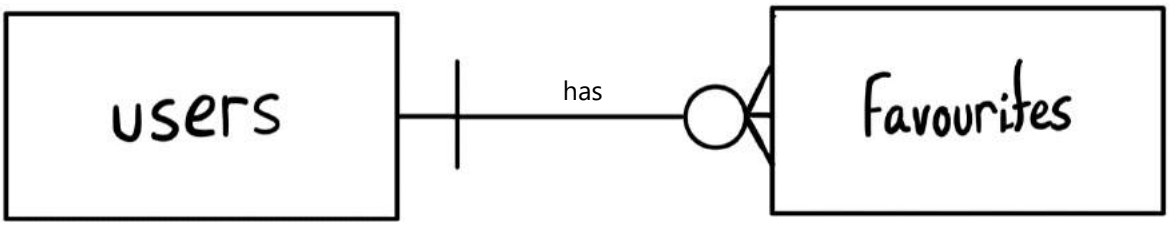
P5	1. Check if the user is logging in and update the page 1.1. IF post['loginType'] == 'Login' THEN 1.1.1. Set element S2 to Layout 2 1.1.2. Set value of element S3 iii. to 'Login' 1.2. END IF 2. Check if the user is resgistering and update the page 2.1. IF post['loginType'] == 'Register'THEN 2.1.1. Set element S2 to Layout 3 2.1.2. Set value of element S3 iii. to 'Register' 2.2. END IF
P6	1. Destroy the session 2. Return to initial page
P7	1. Get the values of the currently selected form and concatenate to get the scale 1.1. SET startNote TO startNote.value 1.2. SET scaleType TO scaleType.value 1.3. SET scale TO concatenate(startNote & scaleType) 2. Initially set the heart icon to hollow and select the default favourite option 2.1. Set element S10 to be a hollow heart icon 2.2. Select the first option for the element S13 3. If the selected scale is a favourite, update the form elements 3.1. REPEAT FOR EACH favourite[ ] 3.1.1. IF scale == favourite[index] THEN 3.1.1.1. Set element S10 to be a filled heart icon 3.1.1.2. Select the option at this index for the element S13 3.1.2. END IF 3.2. END LOOP 4. Update the screen to display the selected scale 4.1. Set element S2 to Layout 1 4.2. DISPLAY scale IN element S3

P8	**1.** Get the scale that is to be favourited **1.1.** SET startNote TO post['startNote'] **1.2.** SET scaleType TO post['scaleType'] **2.** Get all the favourite scales from the database and assume the scale is not favourited **2.1.** SET results[ ] TO Query **Q3** **2.2.** SET favourited TO false **3.** Check if the scale favourited is already in the table **3.1.** REPEAT FOR EACH result[ ] **3.1.1.** IF startNote and scaleType MATCH result[index] **3.1.1.1.** SET favourited TO false **3.1.2.** END IF **3.2.** END LOOP **4.** If the scale was already favourited, delete it from the favourites table **4.1.** IF favourited == true THEN **4.1.1.** Query **Q4** **4.2.** END IF **5.** If it is a new favourite, add it to the favourites table **5.1.** IF favourited == false THEN **5.1.1.** Query **Q5** **5.2.** END IF **6.** Update the favourite button icon, element S10

P9	1. Get the scale that is to be favourited 1.1. SET startNote TO post['startNote'] 1.2. SET scaleType TO post['scaleType'] 2. Get the order of notes to play (allScales is a predefined 2D array of scales) 2.1. SET scale[ ] TO allScales[scaleType] 3. Play the scale, offsetting the scale by the value of the startNote 3.1. REPEAT FOR EACH scale[ ] 3.1.1. SET index TO scale[index] + startNote 3.1.2. Perform output **P11**, passing in index 3.1.3. WAIT 0.5s 3.2. END LOOP
P10	1. Find the startNote and scaleType from the option selected 1.1. SET startNote TO first part of favouritesList.value 1.2. SET scaleType TO second part of favouritesList.value 2. Fill in the form with these values (this will trigger **P7**) 2.1. FILL IN element S8 WITH VALUE startNote 2.2. FILL IN element S9 WITH VALUE scaleType
P11	1. The index of the key will be passed in, play the key 1.1. PLAY AUDIO FILE allSounds[index] 1.2. Animate the key at the index to be pressed

Design of Integration

Entity Relationship Diagram.



ENTITY NAME: users					
Field Name	Key	Type	Size	Required	Validation
username	PK	Varchar	15	✓	Unique Required
password		Varchar	15	✓	Required

SQL QUERY DESIGN

```
CREATE TABLE users (
        username varchar(20) NOT NULL PRIMARY KEY,
        password varchar(20) NOT NULL
);
```

ENTITY NAME: favourites					
Field Name	Key	Type	Size	Required	Validation
username	FK	Varchar	15	✓	Unique Required
startNote		Varchar	15	✓	Required
scaleType		Varchar	15	✓	Required

SQL QUERY DESIGN

```
CREATE TABLE favourites (
        username varchar(20) NOT NULL,
        startNote varchar(10) NOT NULL,
        scaleType varchar(10) NOT NULL,
        FOREIGN KEY (username) REFERENCES users(username)
);
```

INDEX	QUERY DESIGN	PARAMETERS
Q1.	DESCRIPTION	To retrieve the password linked to a username. Returns null if there is no matching username.
	QUERY TYPE	SELECT
	FIELDS	password
	TABLE	users
	CRITERIA	WHERE username LIKE '$username'
	SORT ORDER	None
Q2.	DESCRIPTION	To insert a new user into the 'users' table. Used to register a user.
	QUERY TYPE	INSERT
	FIELDS	username , password
	TABLE	users
	CRITERIA	VALUES '$username' , '$password'
	SORT ORDER	None
Q3.	DESCRIPTION	To retrieve all the favourites from the 'favourites' table. Every favourite is made up of a starting note and a scale type, so both need to be retrieved. Only retrieve the records with a matching username. Returns null if there is no matching username.
	QUERY TYPE	SELECT
	FIELDS	startNote , scaleType
	TABLE	favourites
	CRITERIA	WHERE username LIKE '$username'
	SORT ORDER	None
Q4.	DESCRIPTION	To delete a favourite scale. This will remove the record containing one of a user's favourites.
	QUERY TYPE	DELETE
	FIELDS	*
	TABLE	favourites
	CRITERIA	WHERE startNote = '$startNote' AND scaleType = '$scaleType' AND username LIKE '$username'
	SORT ORDER	None

Q5.	DESCRIPTION	To add a new favourite. Updates the table to include a new favourite scale for a user.
	QUERY TYPE	INSERT
	FIELDS	username
	TABLE	favourites
	CRITERIA	WHERE startNote = '$startNote' AND scaleType = '$scaleType' AND username LIKE '$username'
	SORT ORDER	None
Q6.	DESCRIPTION	To collect all of one user's favourites. Used to fill the options in the 'Choose favourite' element
	QUERY TYPE	SELECT
	FIELDS	*
	TABLE	favourites
	CRITERIA	WHERE username LIKE '$username'
	SORT ORDER	None

Implementation

This section will document the final website. I will include screenshots of the final page and a link to a video to capture the audible part of my website. I will also include all the code for the website. In the internal commentary of my code, I will point out where my code includes advanced higher concepts and where I have implemented integration.

PREFIXES IN THIS SECTION	**DESCRIPTION**

Q
Internal Commentary:
`--------------------`
`Q : Purpose of query`
`--------------------`
	Used to identify the implementation of a query to integrate with the database

AHC
Internal Commentary:
`------------------------------`
`AHC : Description of AH concept`
`------------------------------`
	Used to identify the implementation of an Advanced Higher concept

Fully Implemented Website

My website relies heavily on vision and hearing. Attached here is a practical demonstration of my website, please watch with sound on:

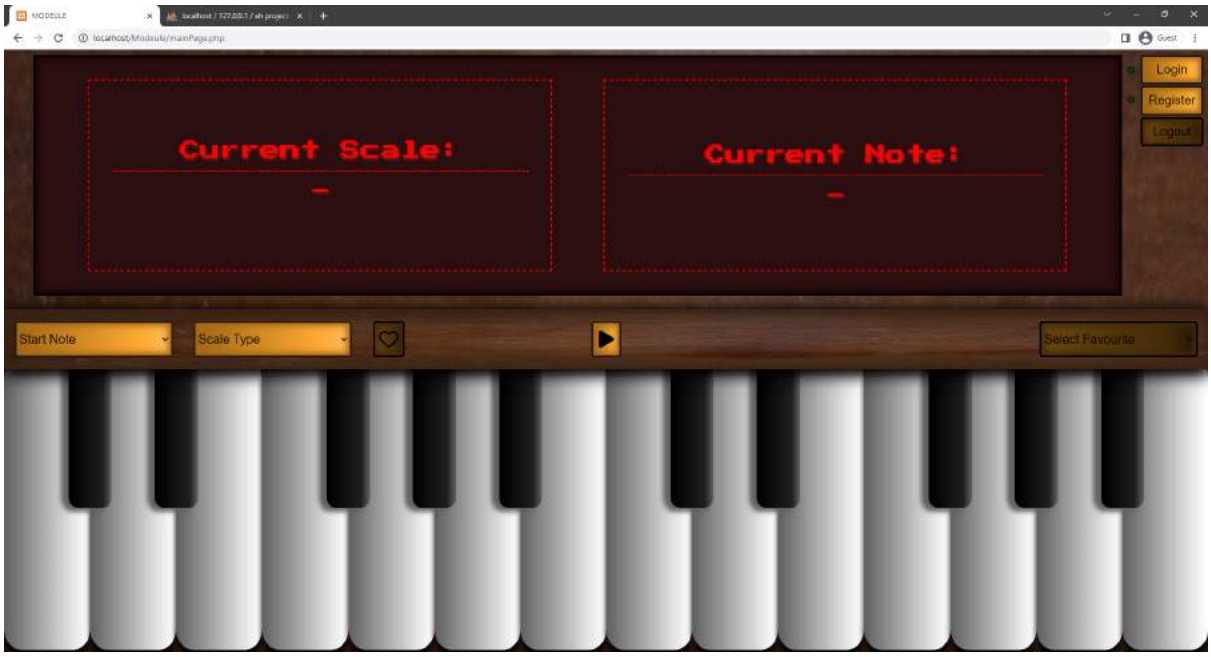
https://www.youtube.com/watch?v=1-Fqnq-HtUw

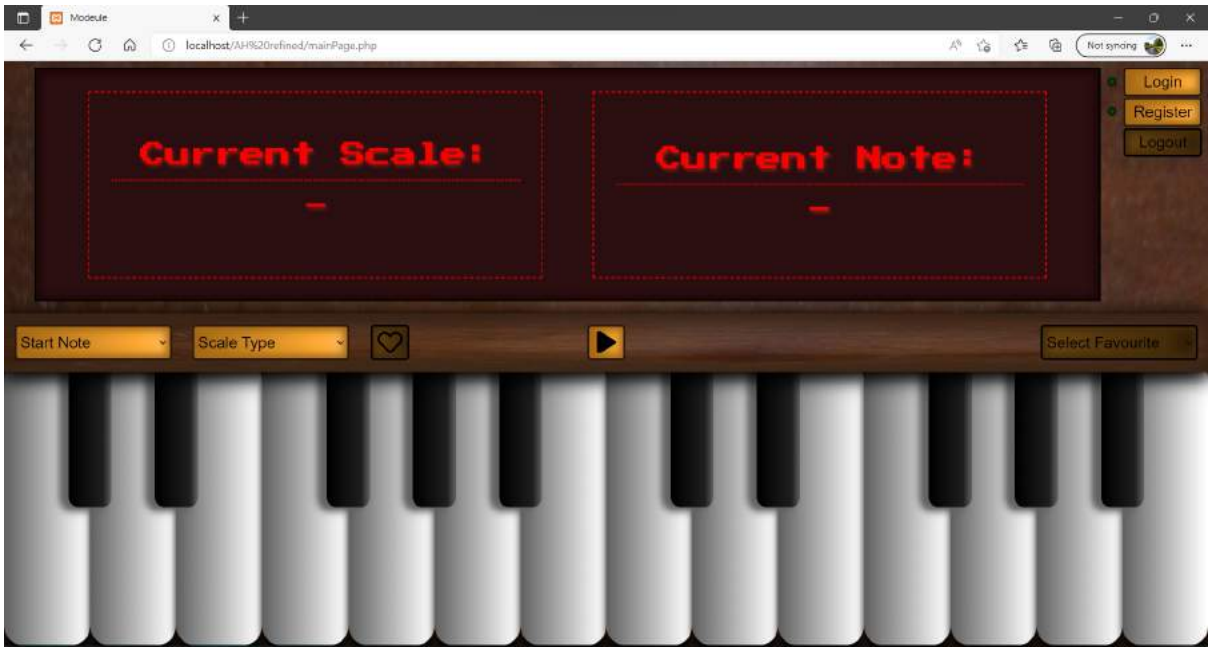
My page is skeuomorphic and imitates a synth. This means it doesn't change layout much, so my website only contains one page. There are parts that change dramatically, these are contained in an element that imitates a digital screen (element S2). JavaScript is used to make this screen change appropriately. JavaScript is also used to disable and enable appropriate inputs on the synth.

Final Product (width > 720px)

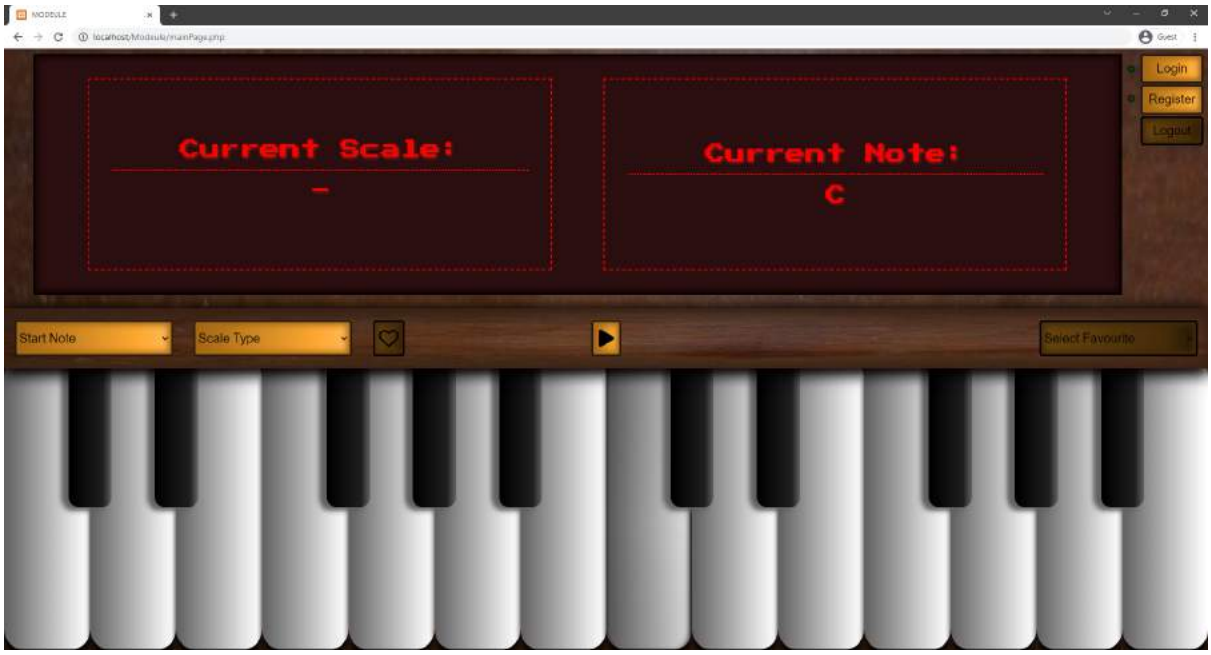
Evidence will be recorded from both Chrome, and Microsoft Edge. The results will be recorded for both browsers for the following situations.

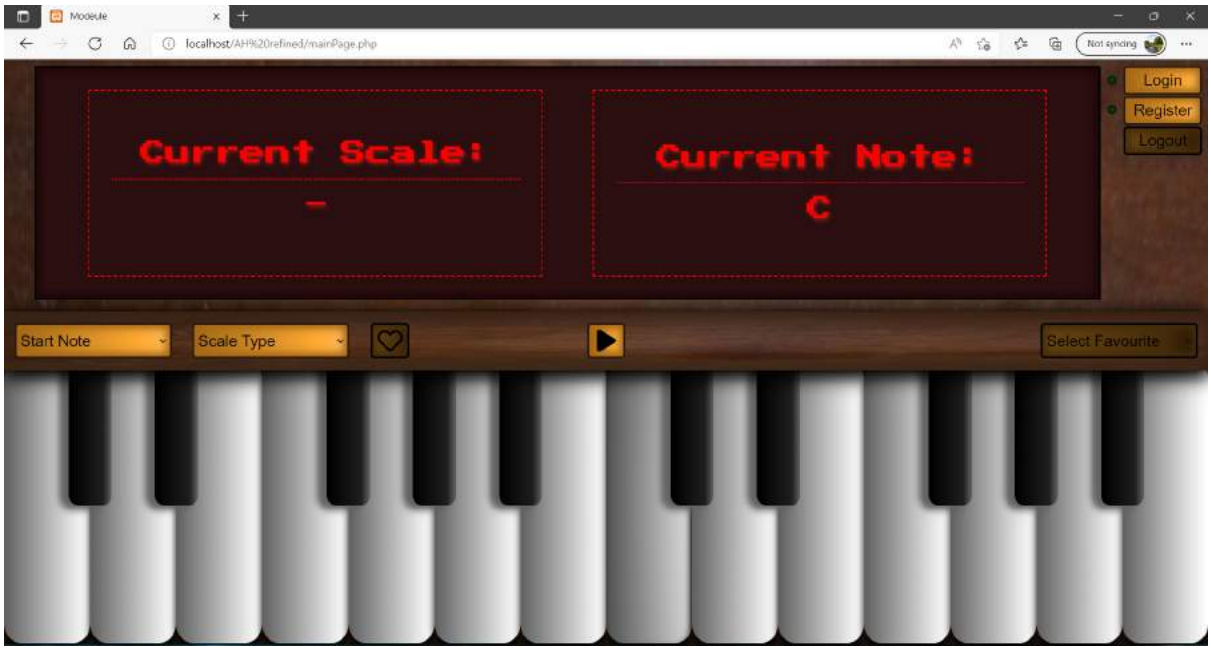
Initial state of page as viewed in chrome at resolution 1920 x 1080:



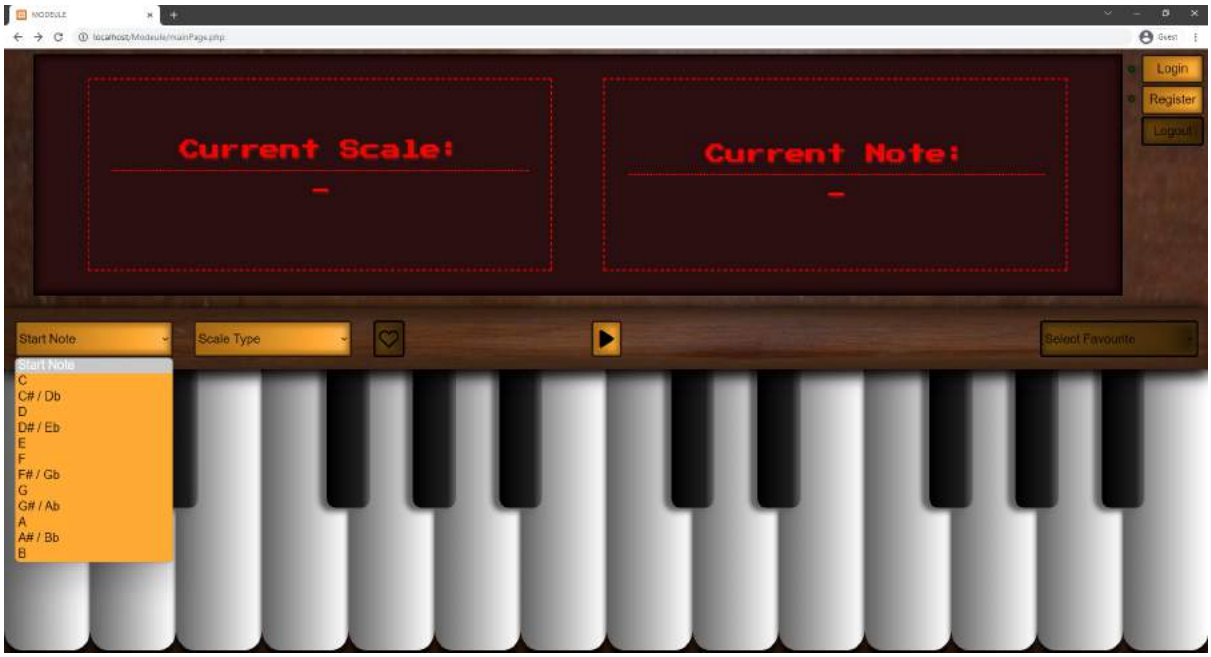


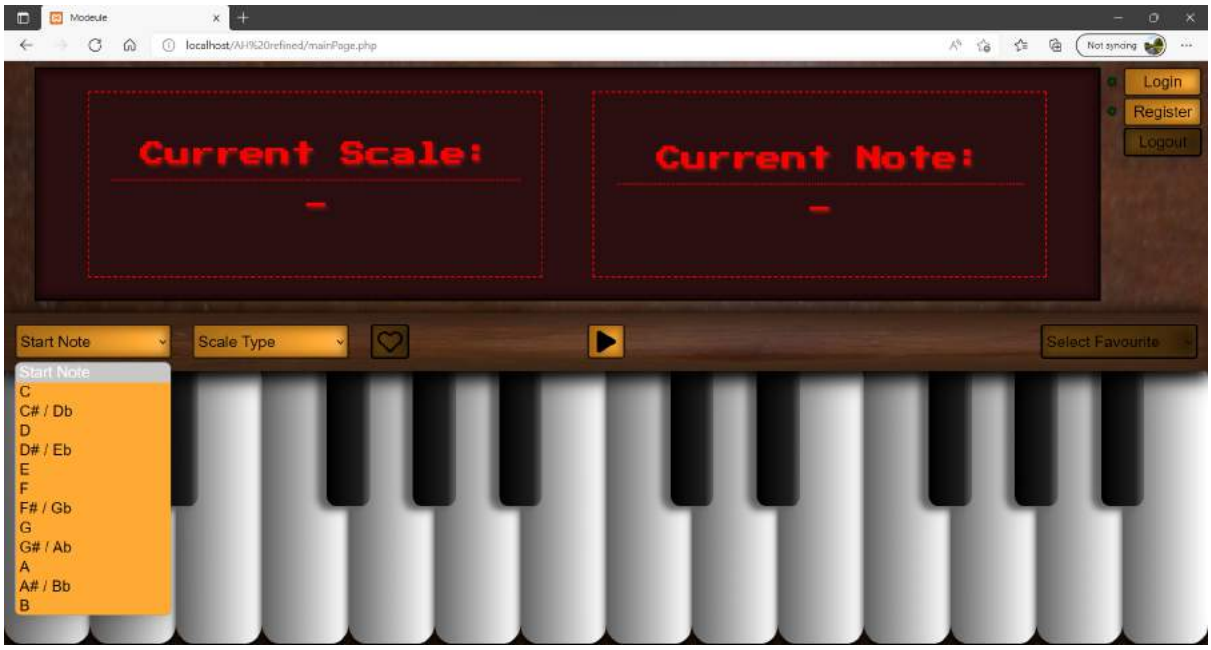
When a key (C) is clicked:



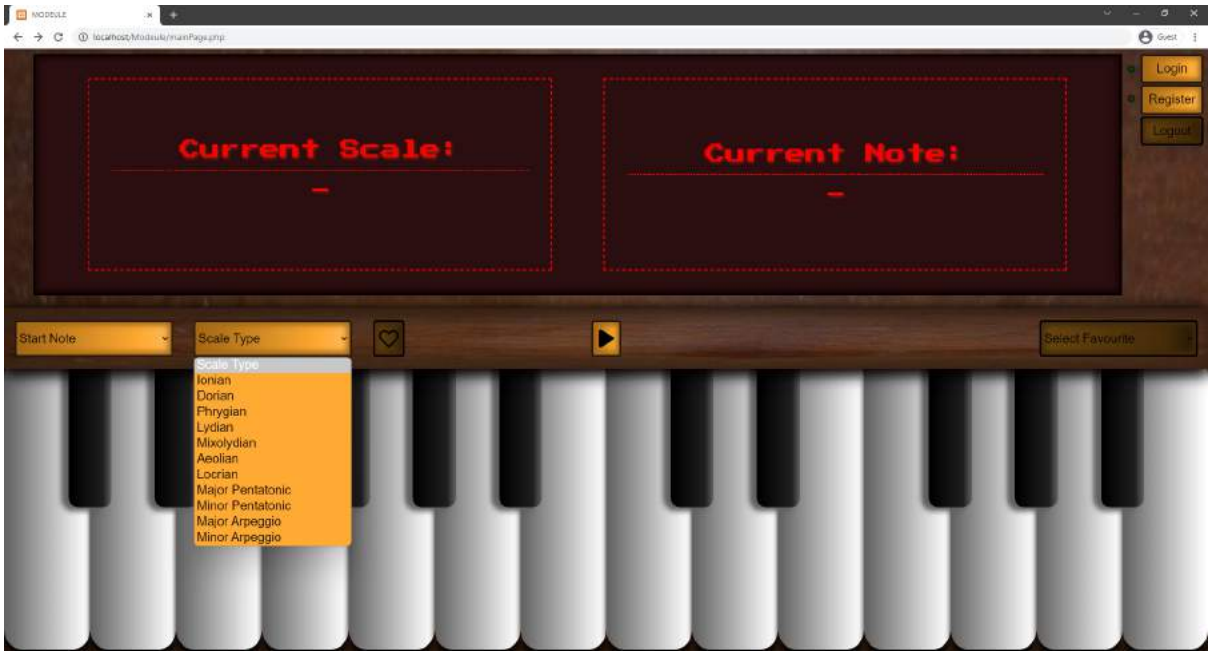


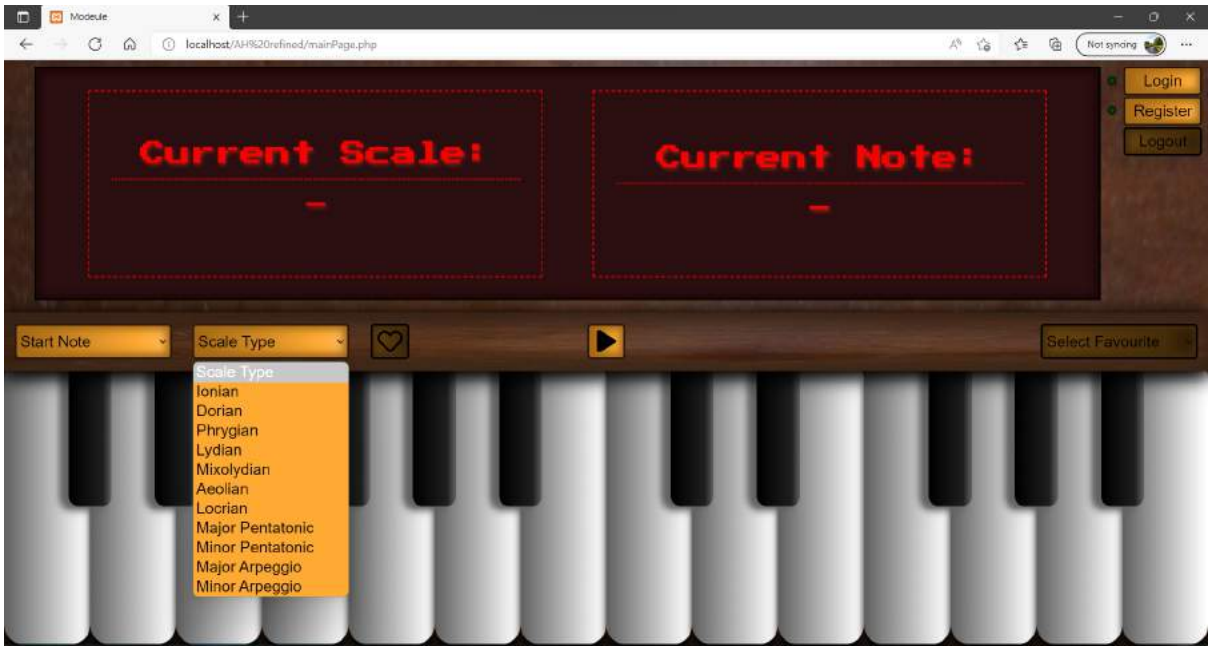
When ‘Start Note’ is clicked:



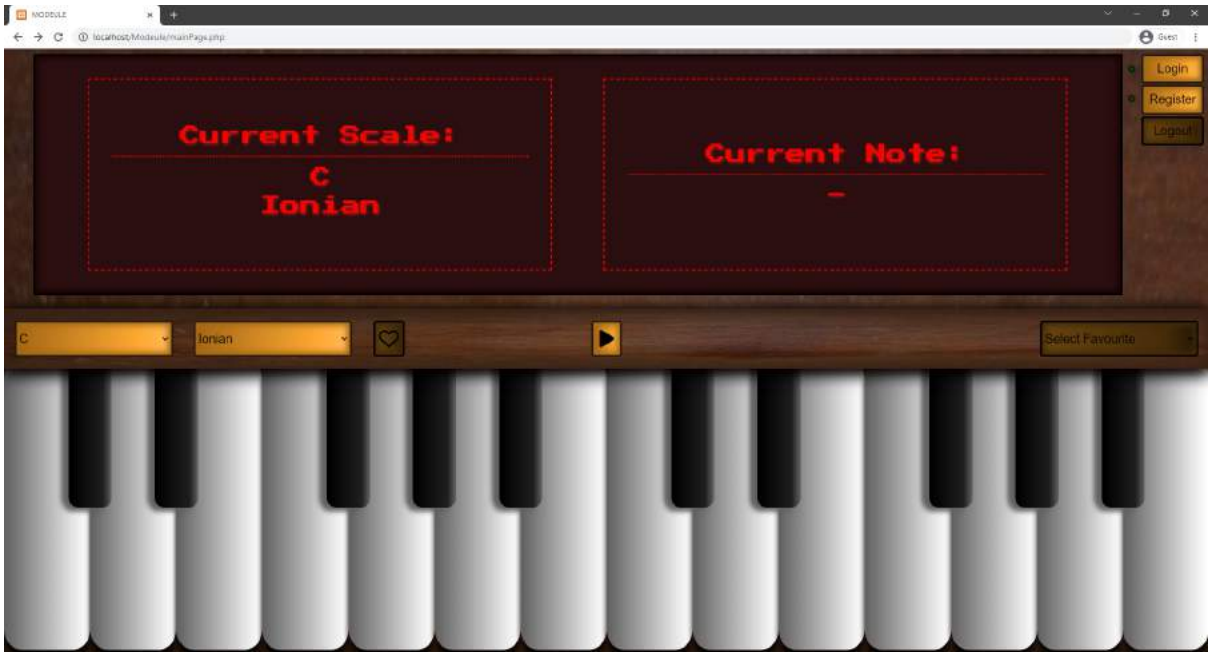


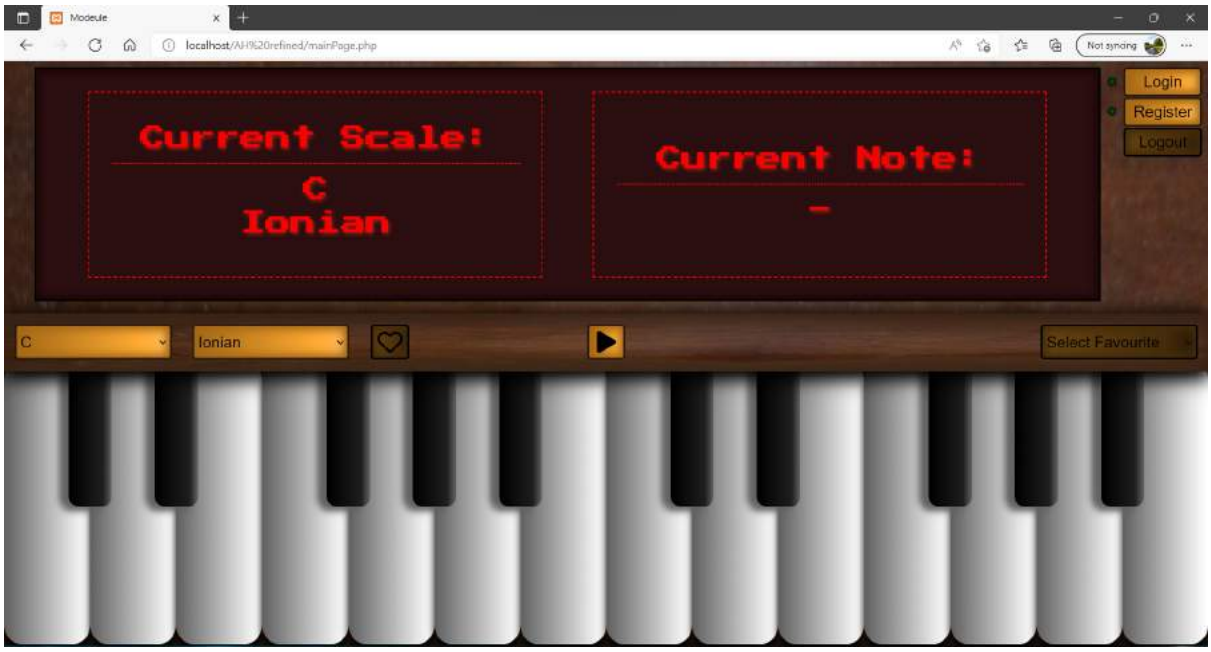
When ‘Scale Type’ is clicked:



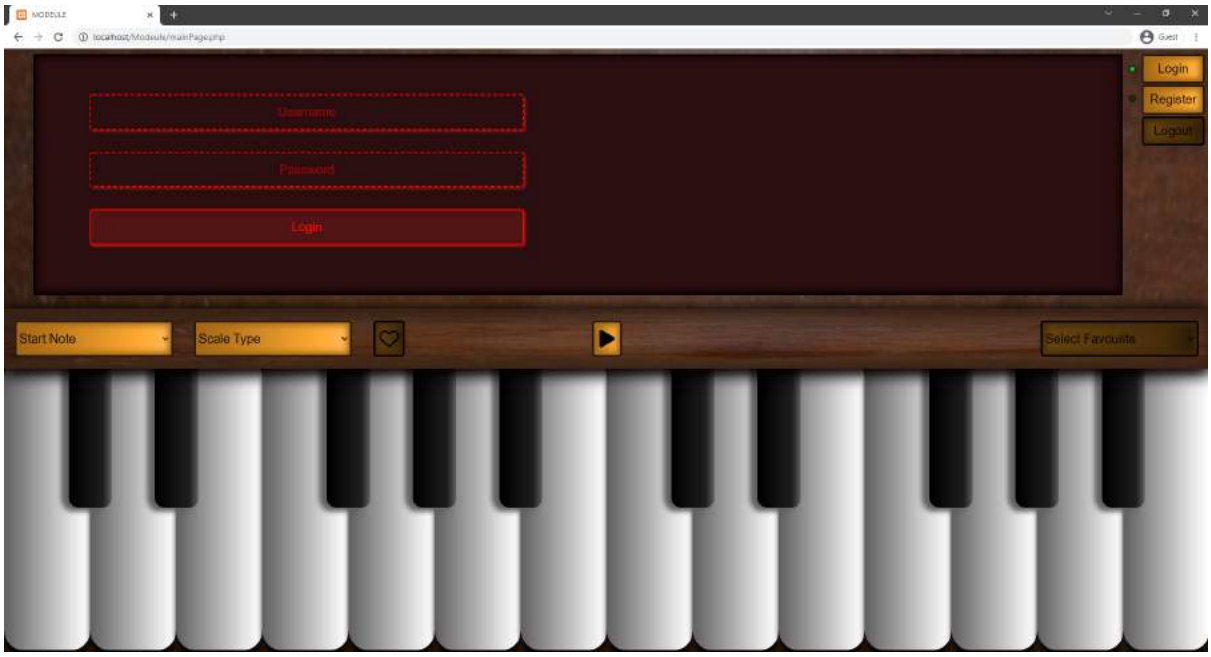


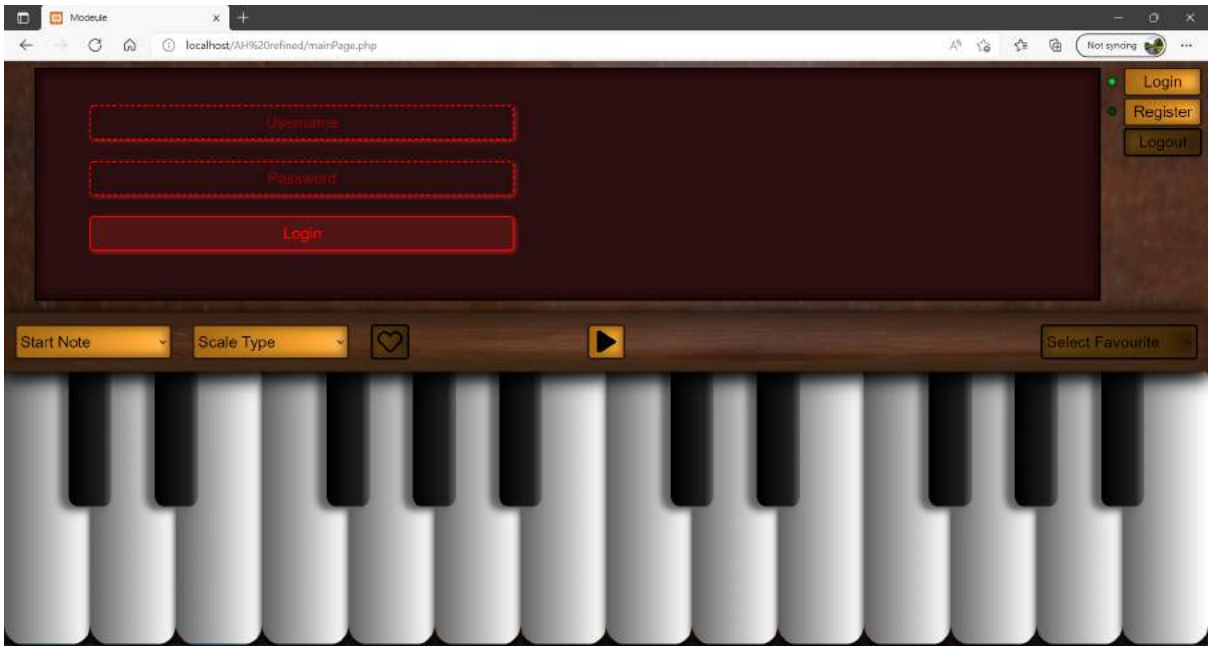
When ‘Start Note’ and ‘Scale Type’ are filled in with values ‘C’ and ‘Ionian’ respectively:



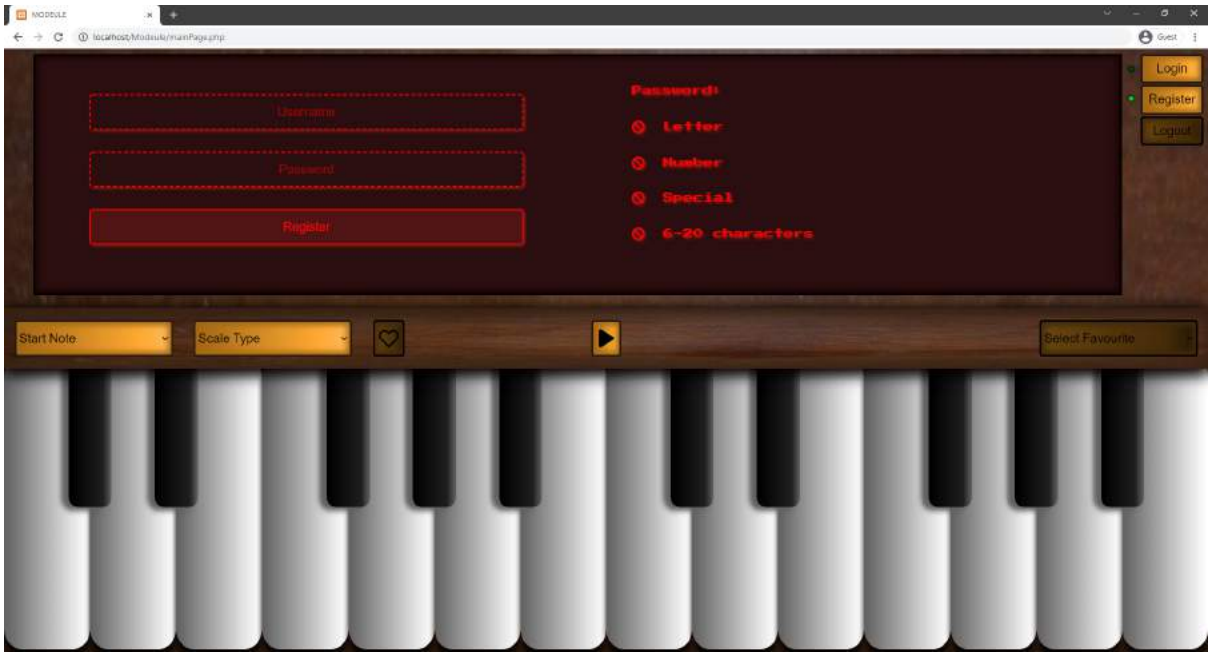


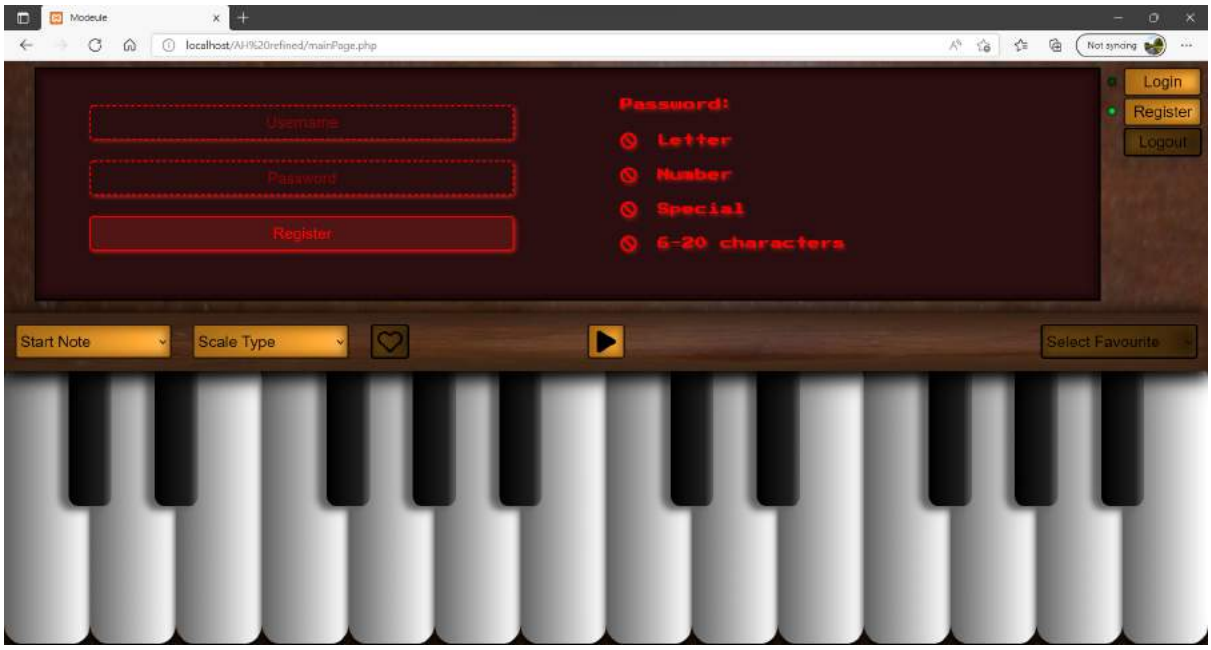
When “Login” button is clicked:



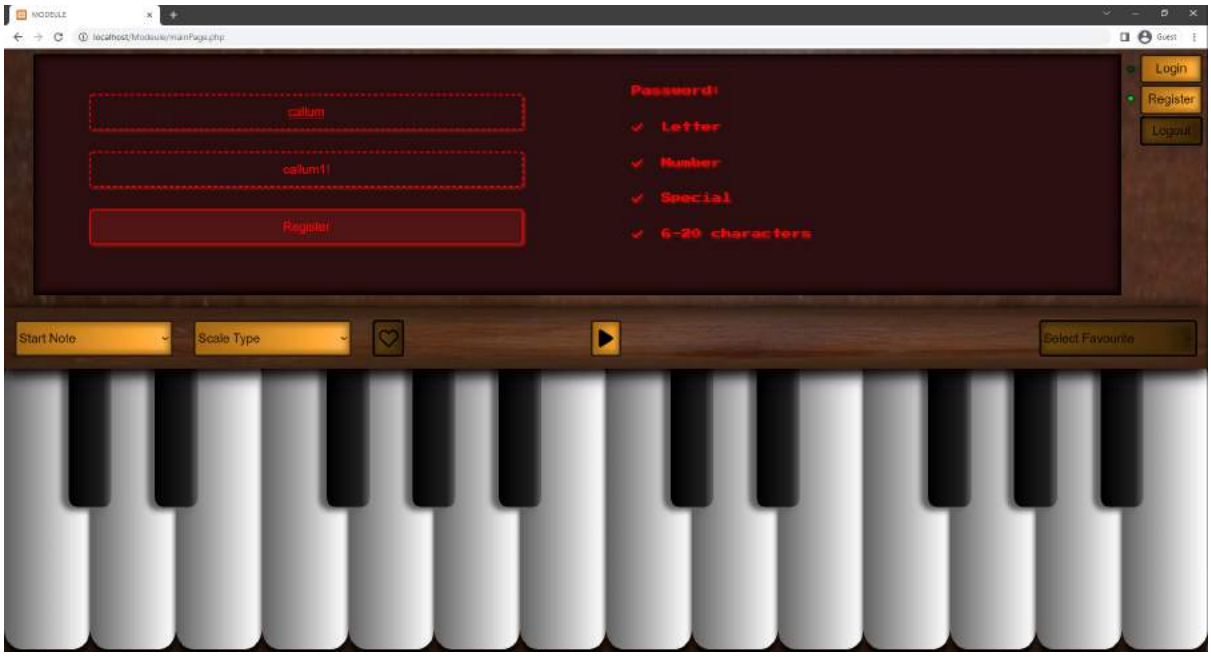


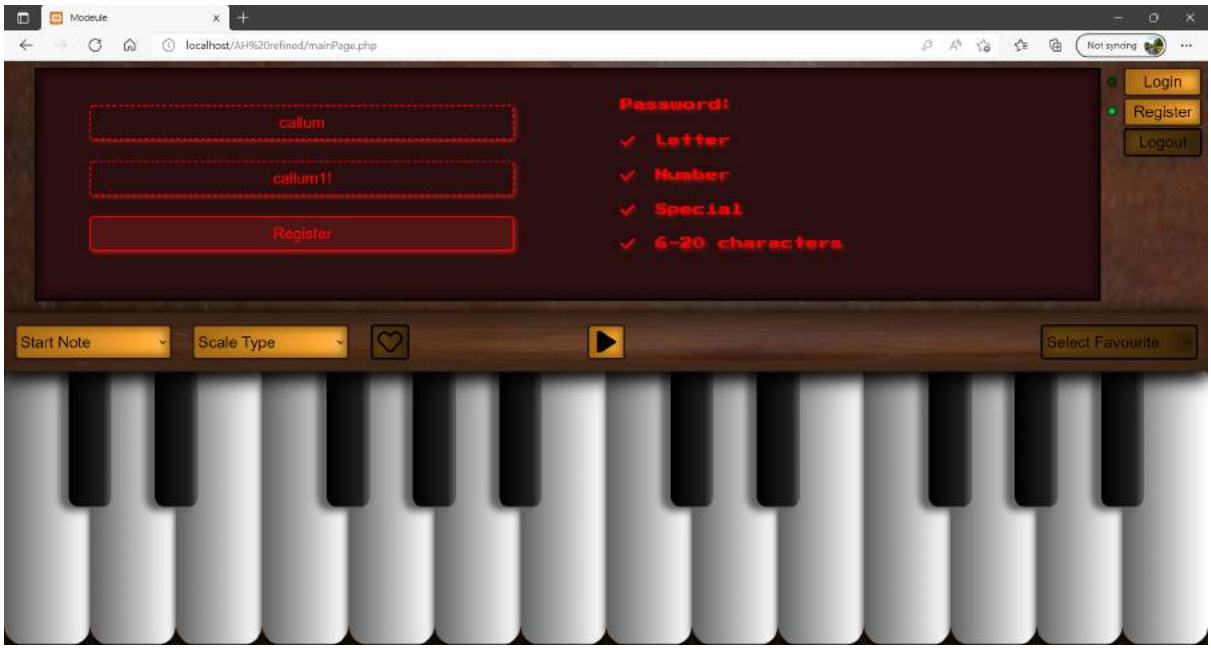
When “Register” button is clicked:





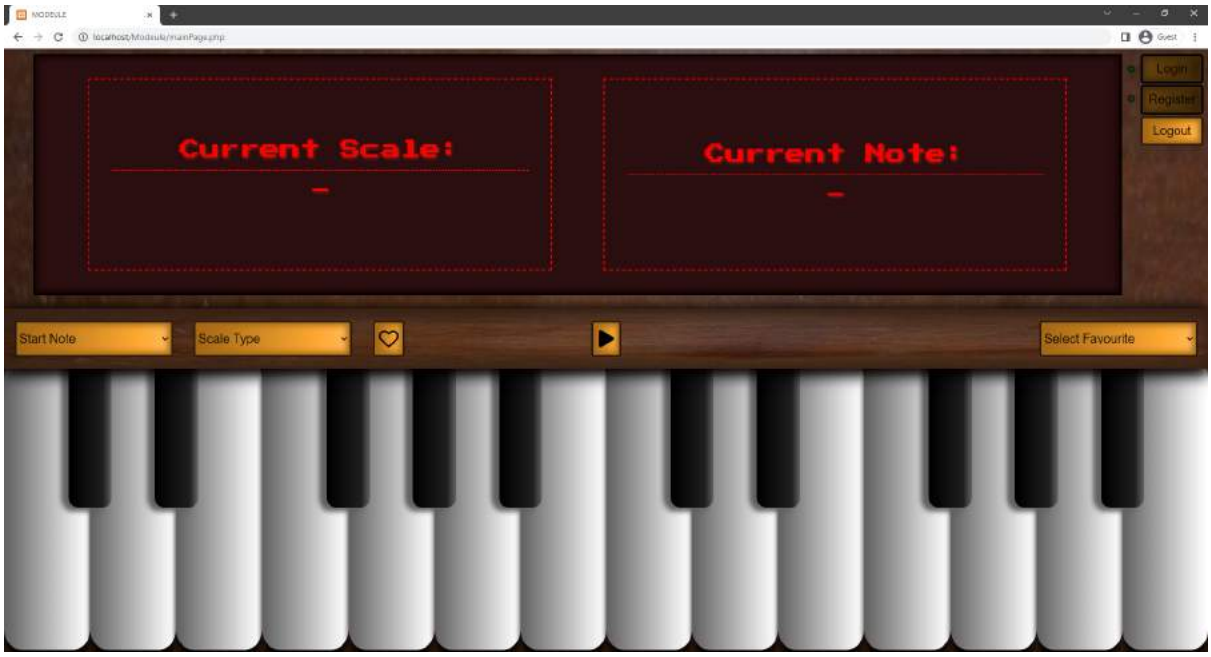
When “Register” button is clicked, and a valid username and password have been entered:

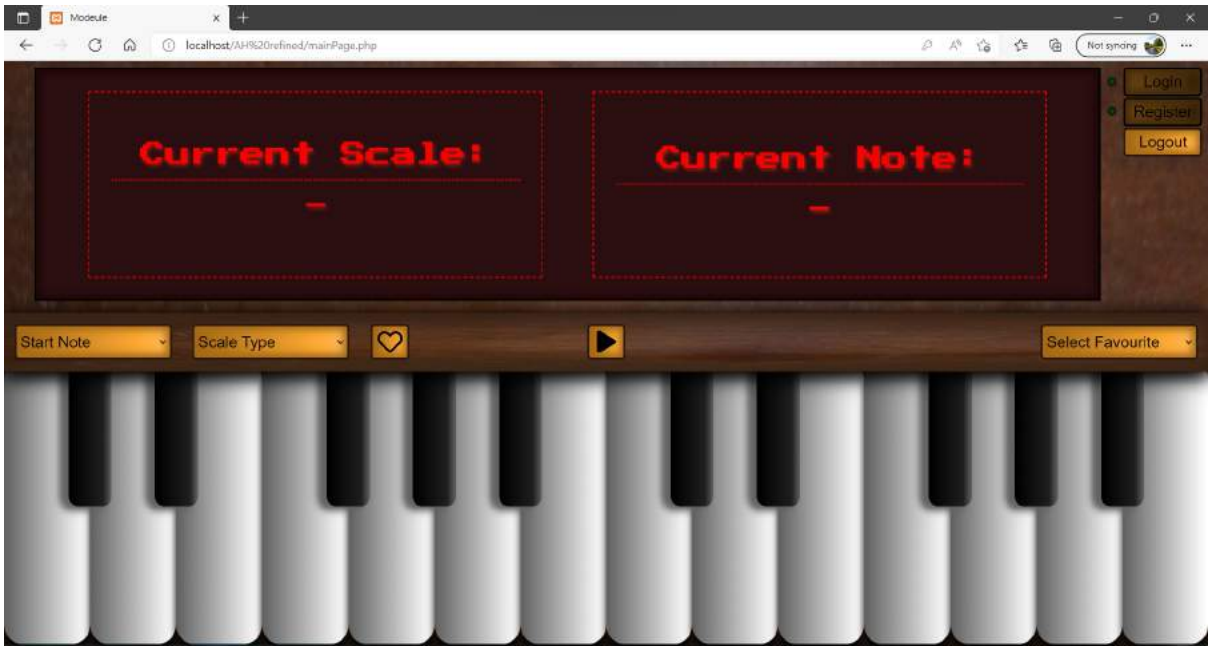




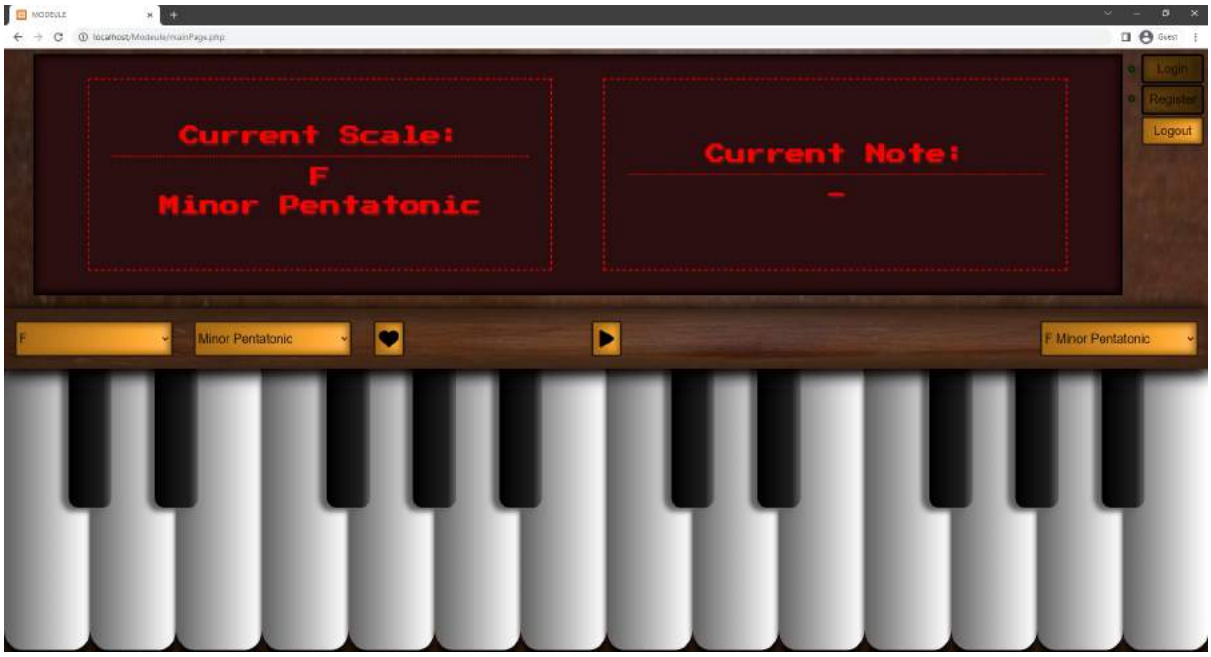
Password is only visible when mouse hovers over it:

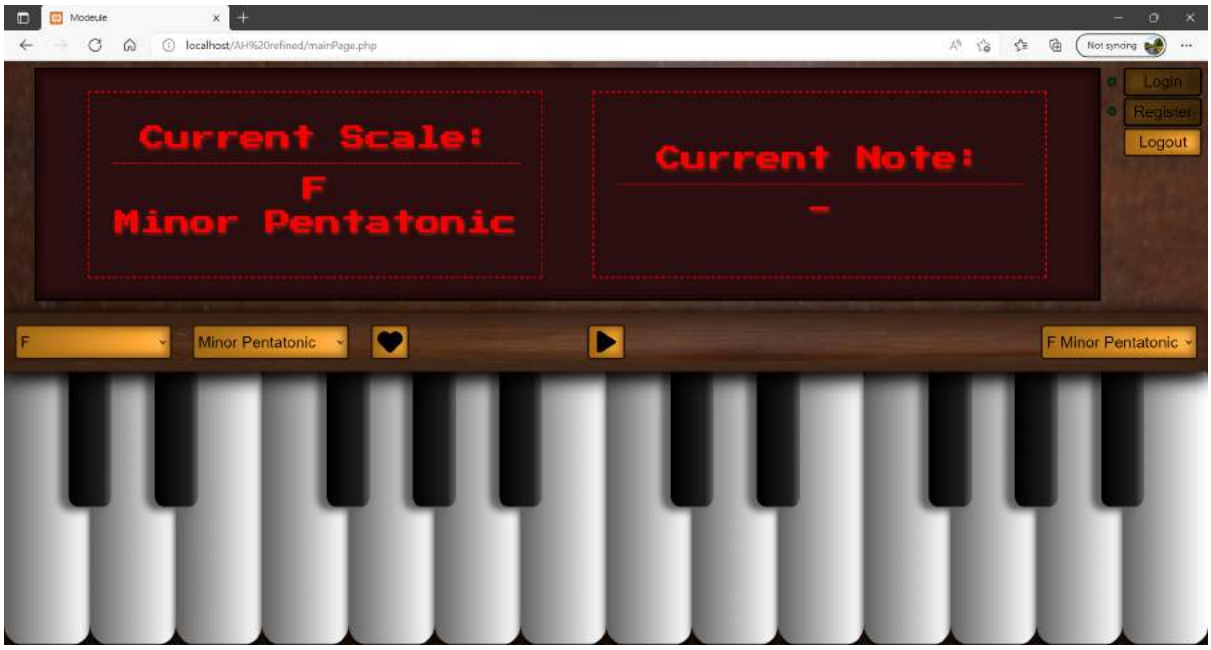
When a user is logged in:



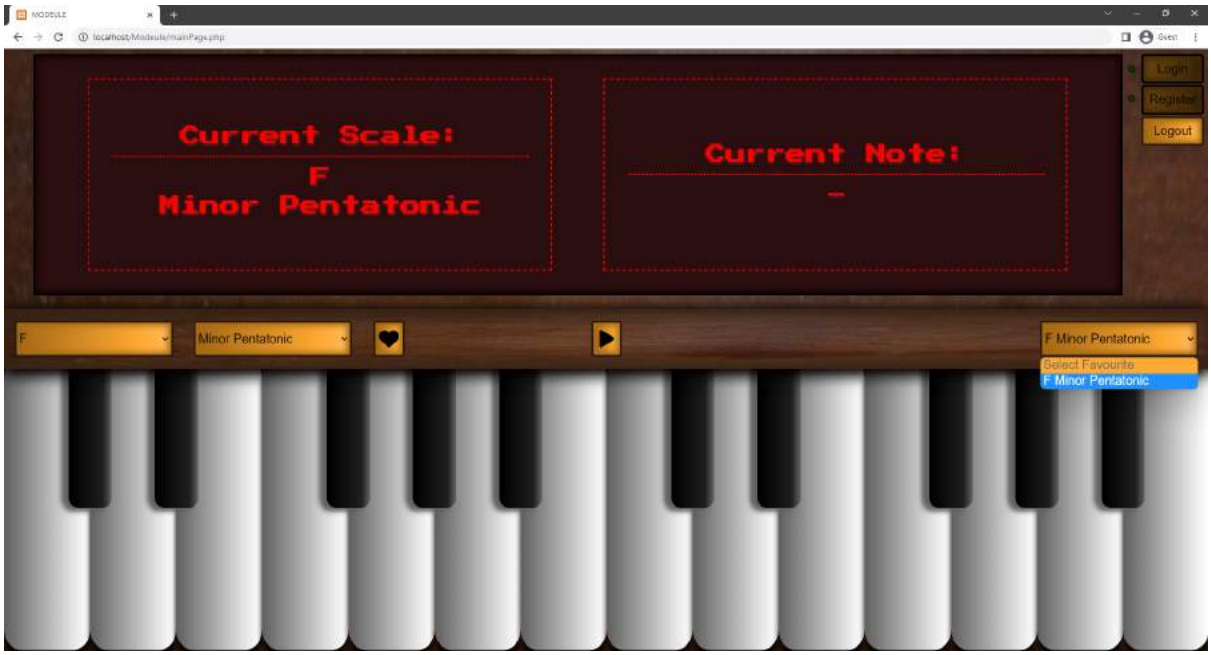


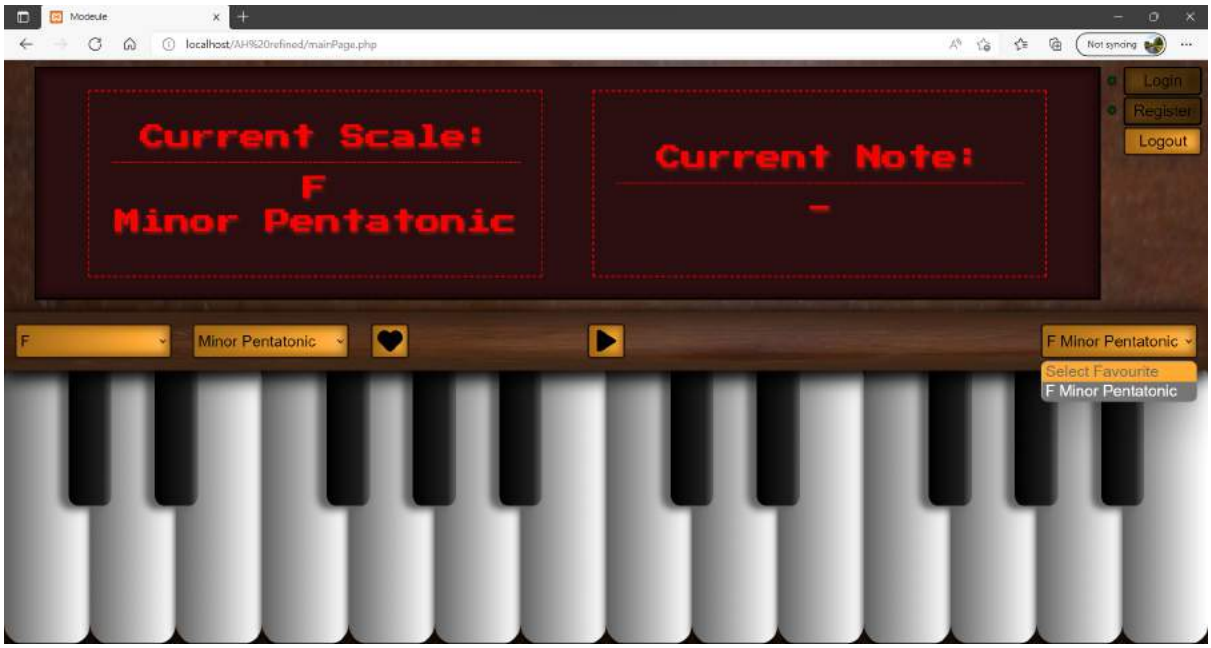
When a logged-in user fills in the scale form, and clicks the ‘Favourite’ button:



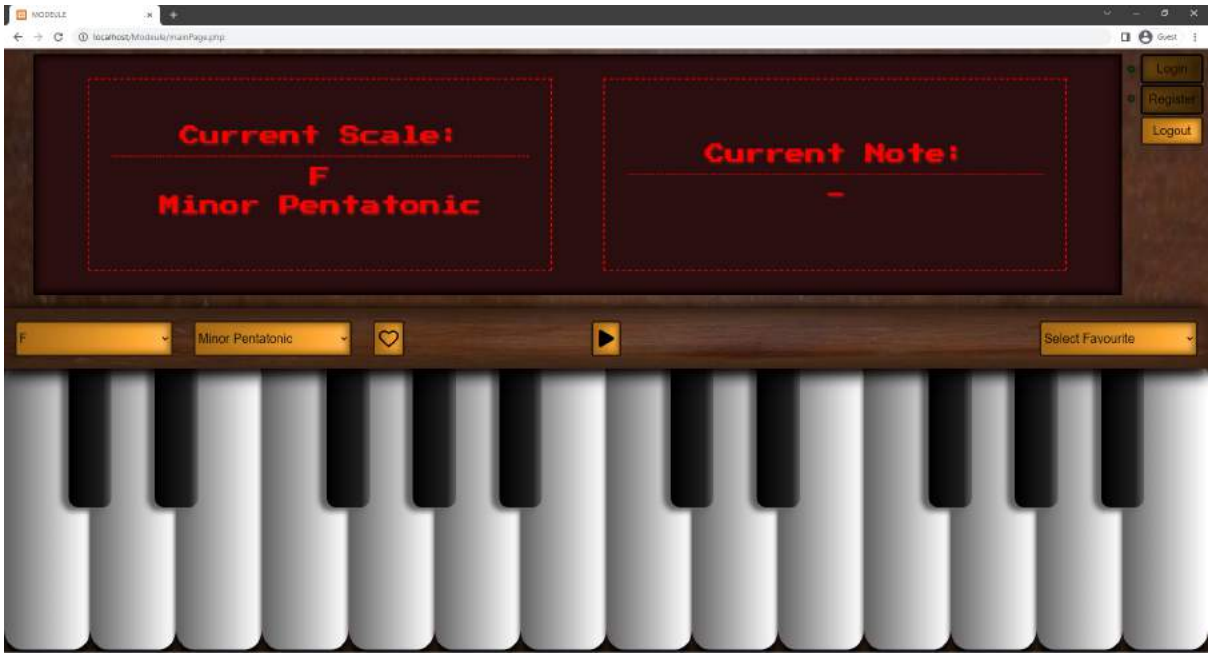


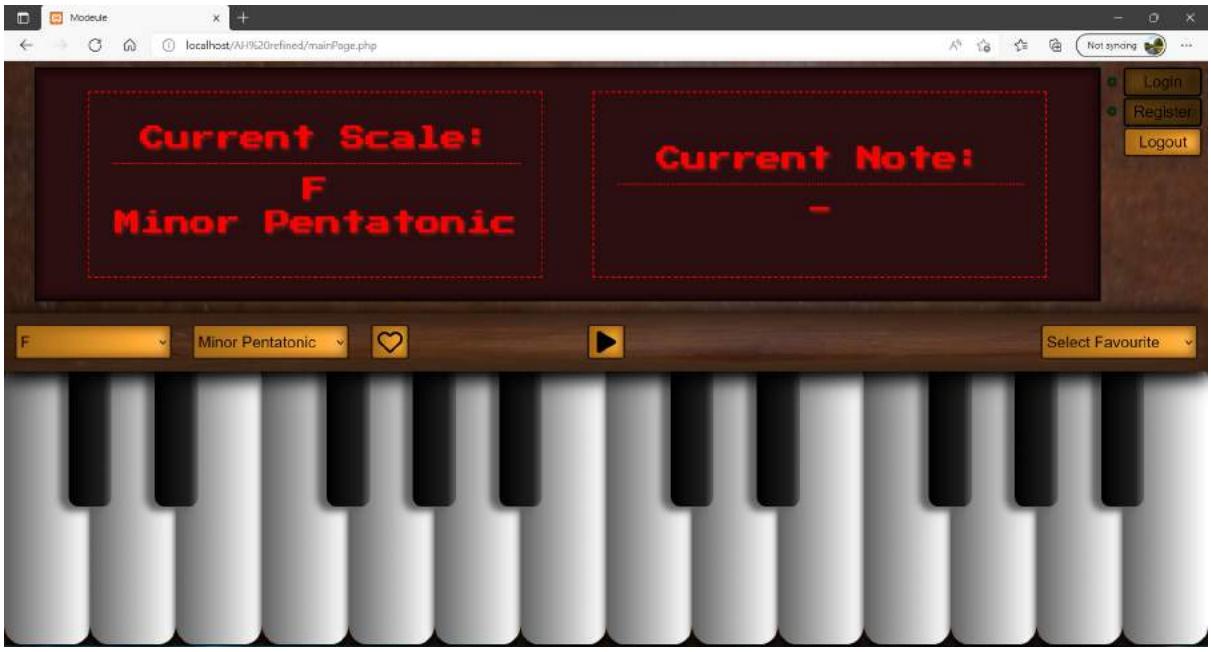
When a logged-in user clicks the ‘Favourites’ dropdown list:



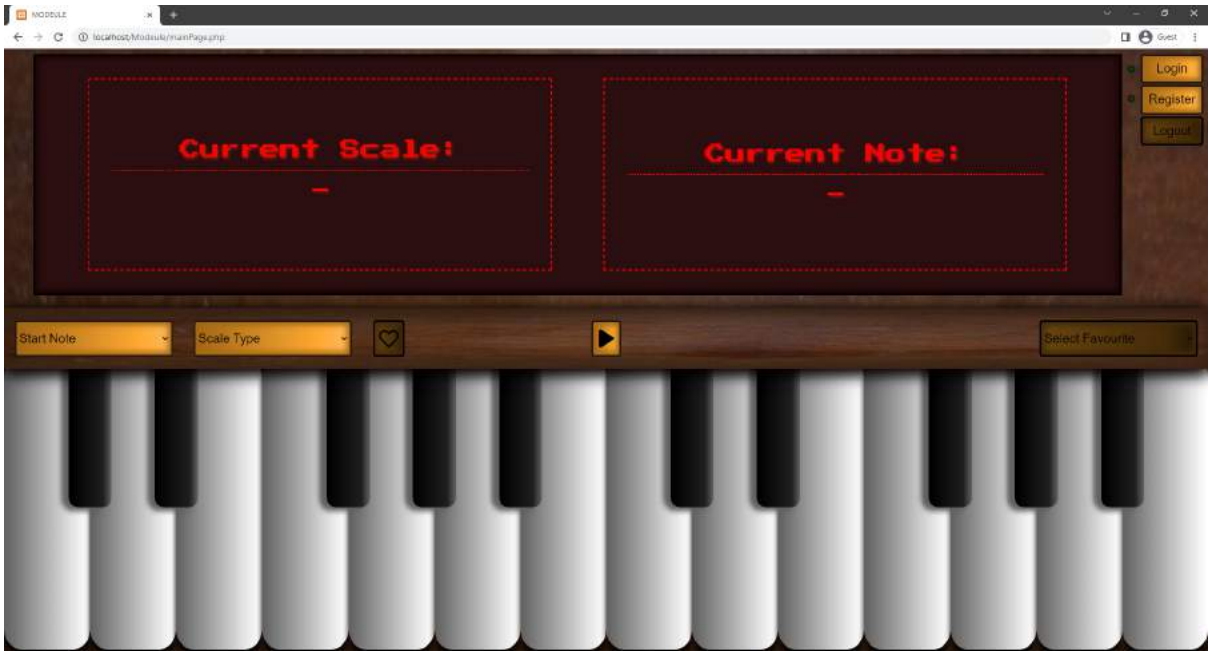


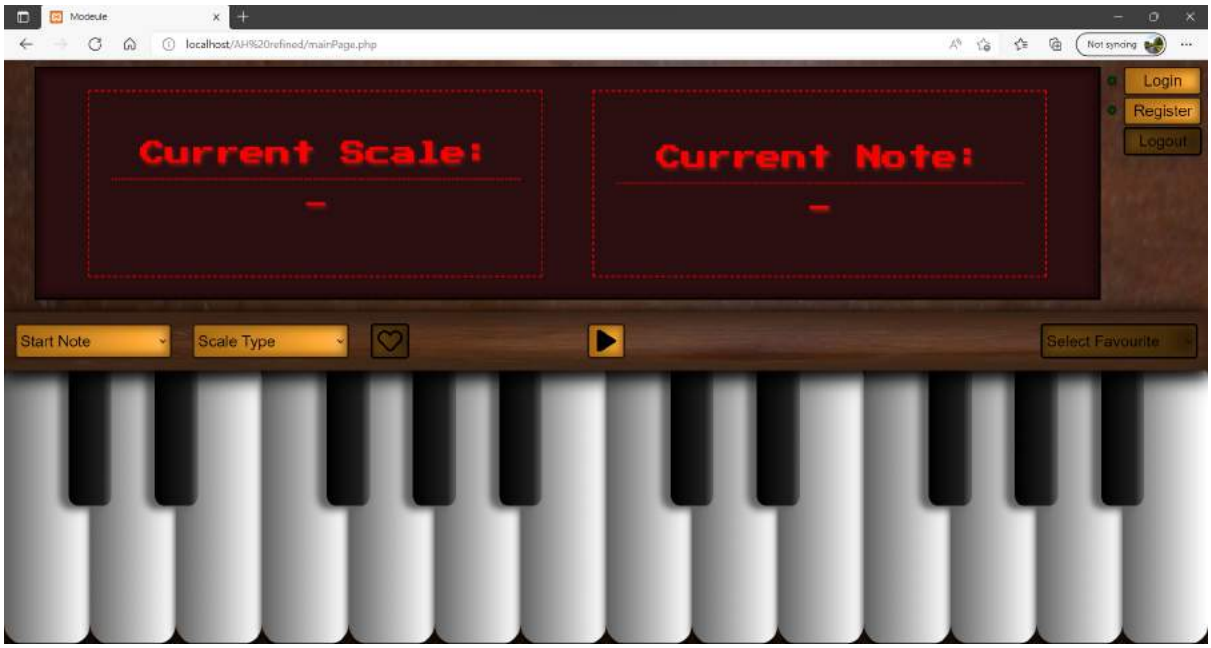
When a logged-in user clicks the ‘Favourite’ button when a favourite is already selected:





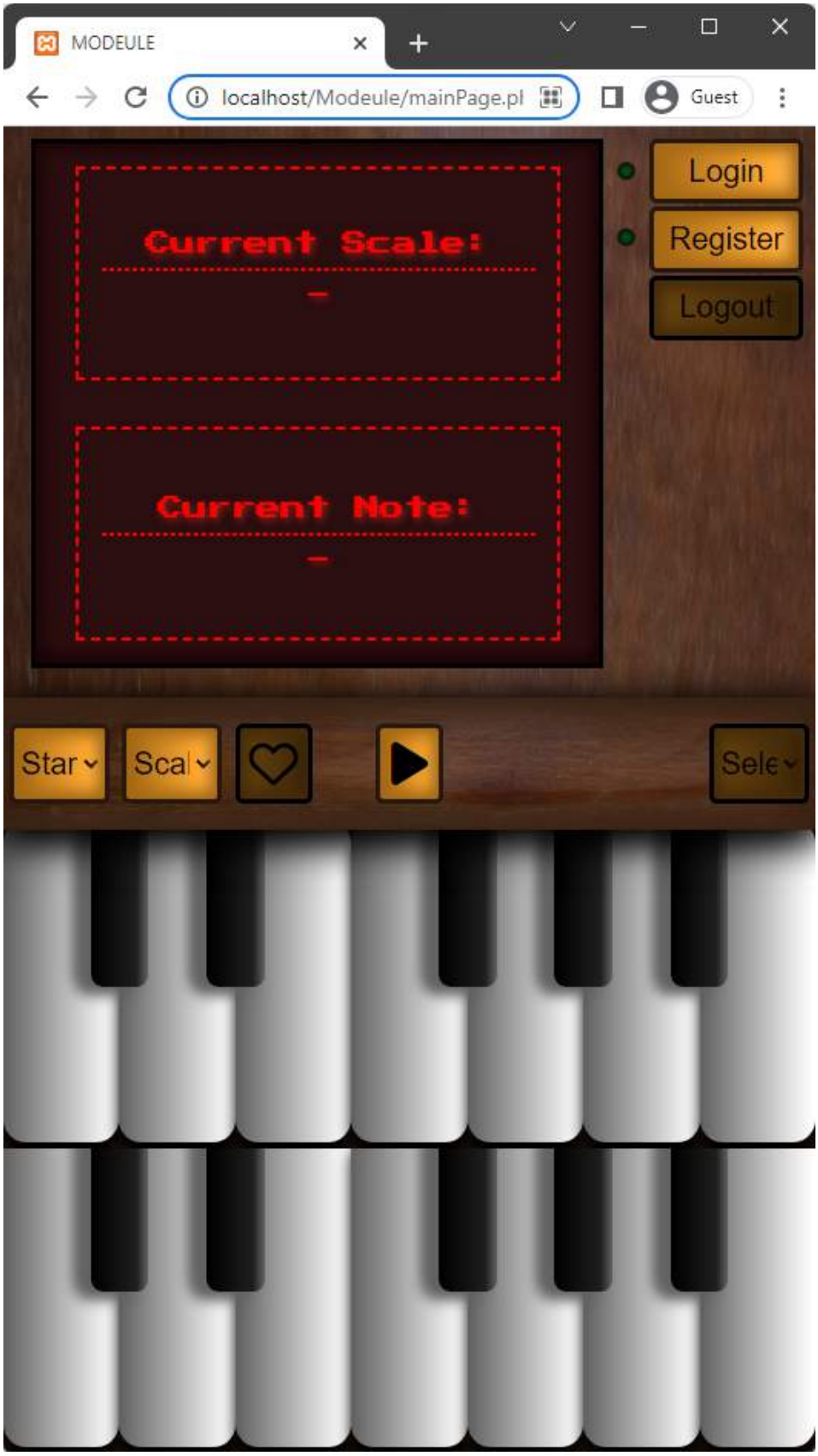
When a logged-in user clicks the ‘Logout’ button:



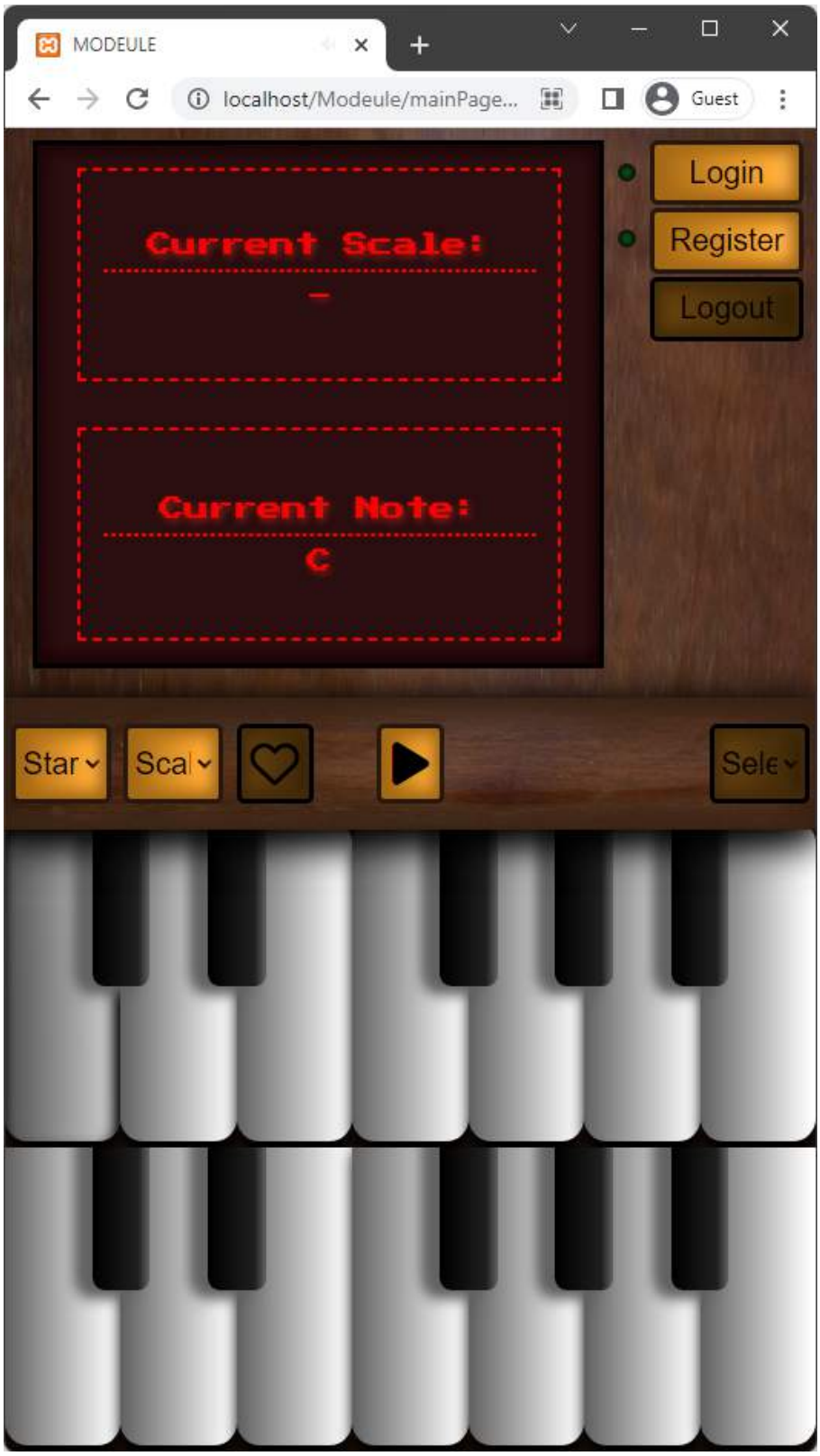


Final Product (width < 720px)

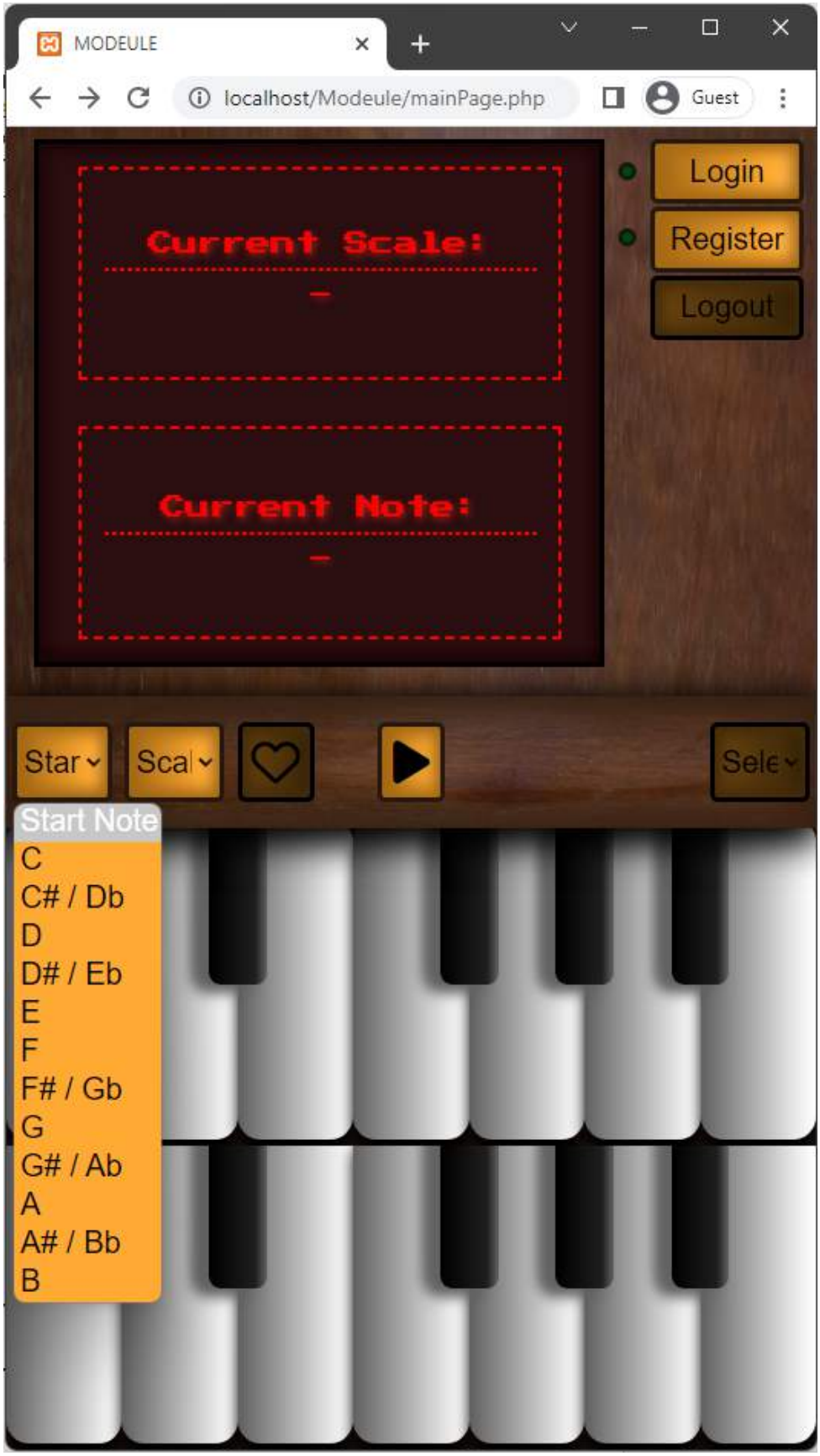
Media queries: here is the site viewed in a browser less than 720px wide:



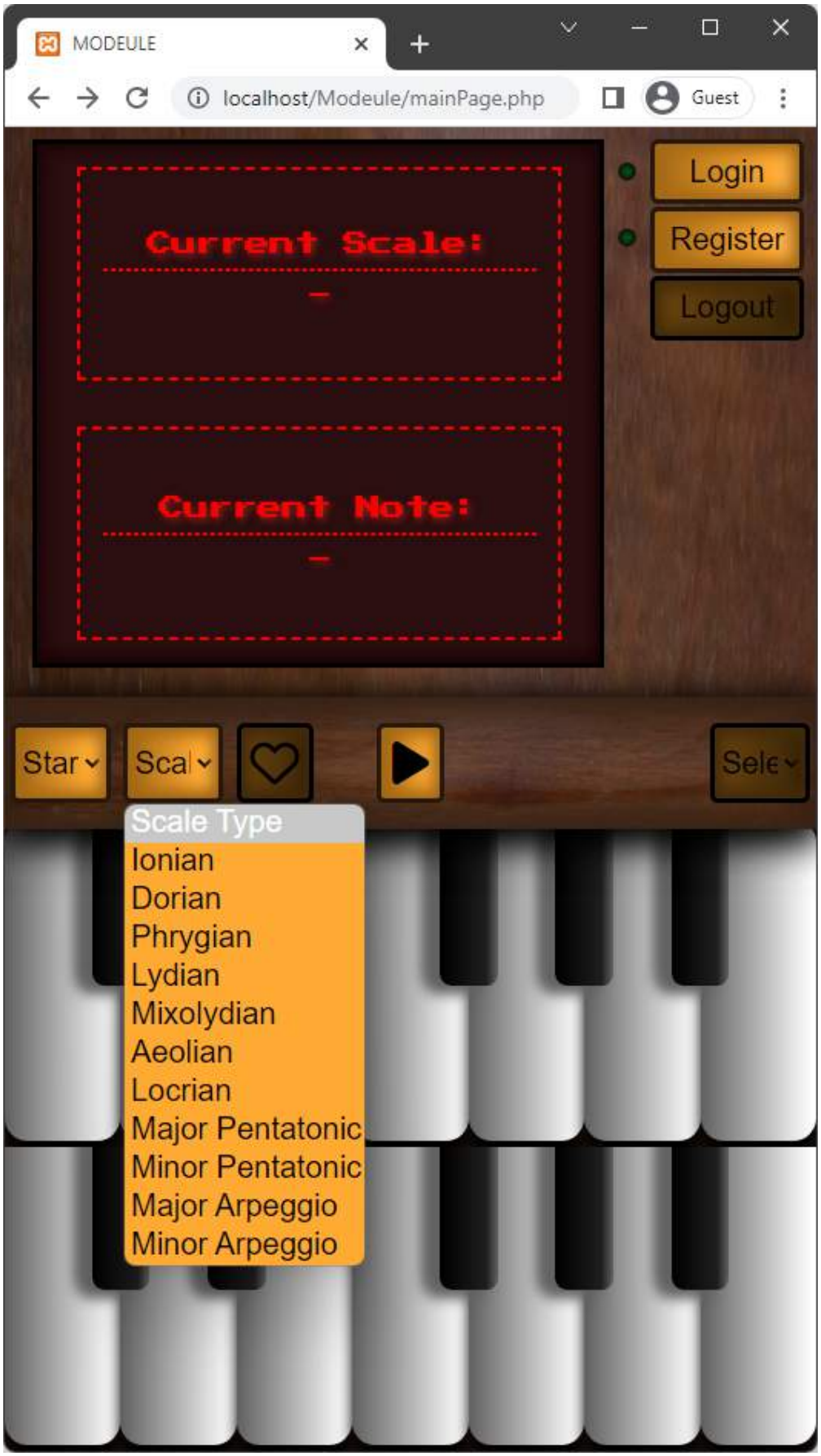
When a key (C) is clicked:



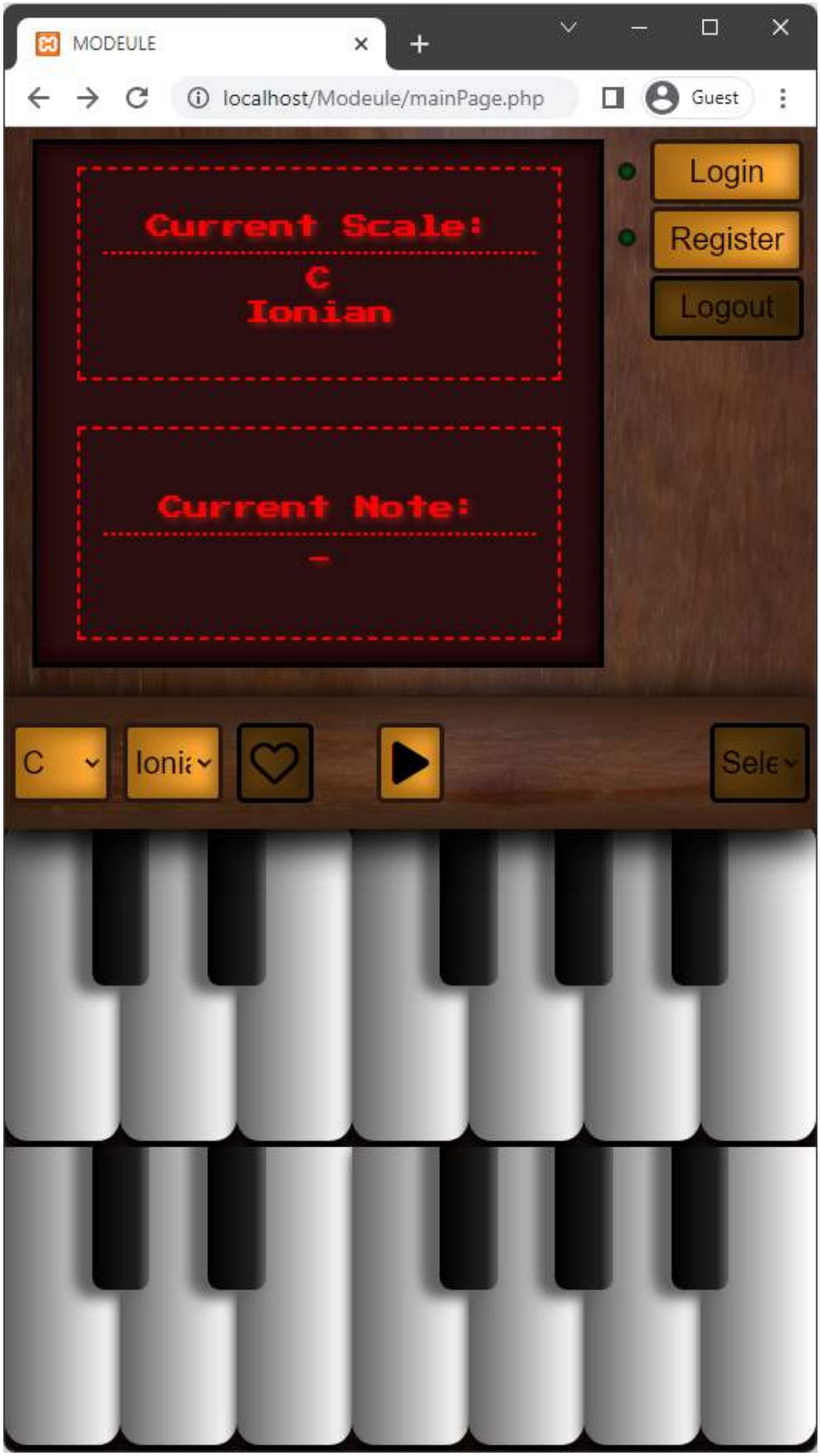
When ‘Start Note’ is clicked:



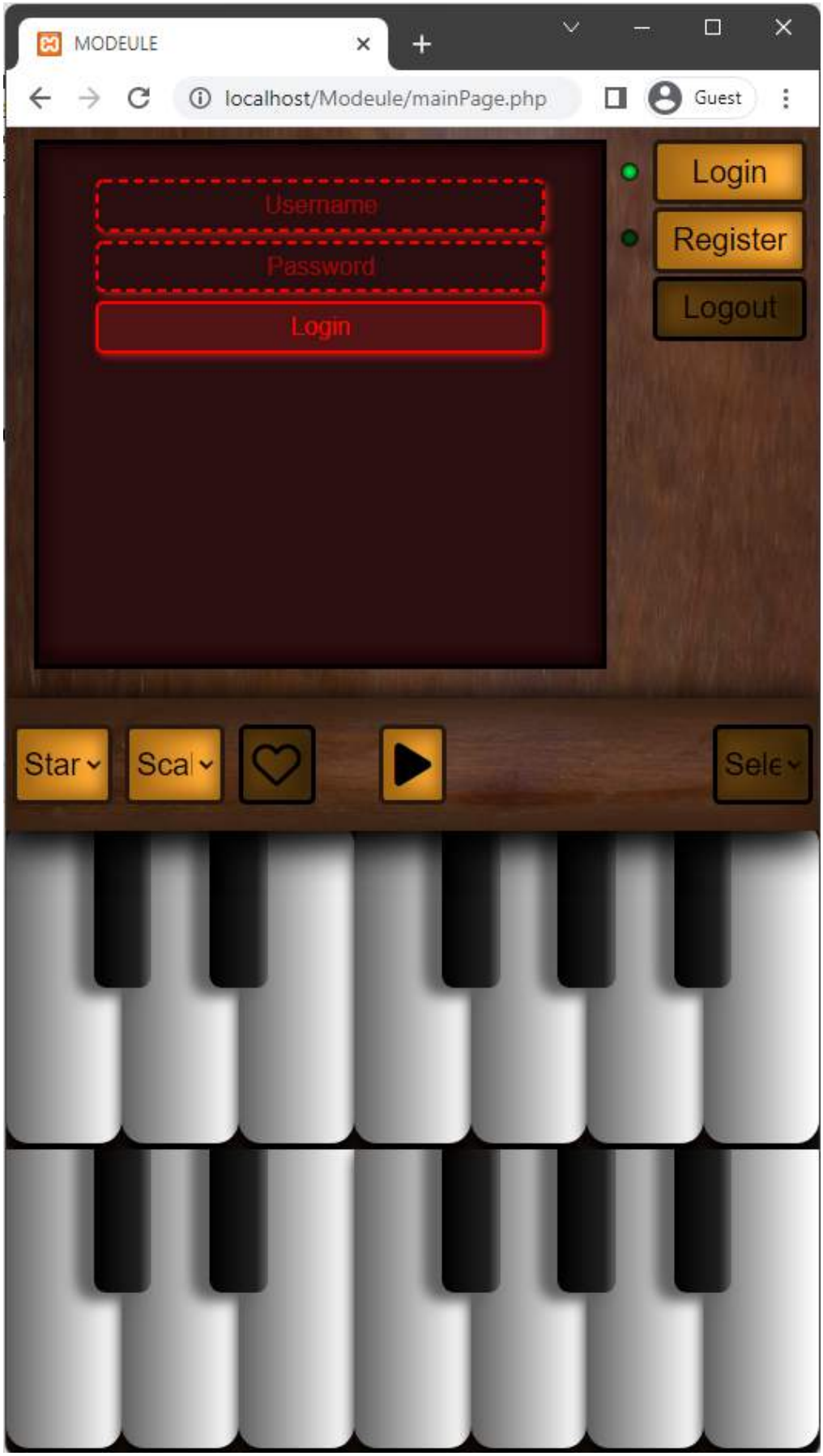
When ‘Scale Type’ is clicked:



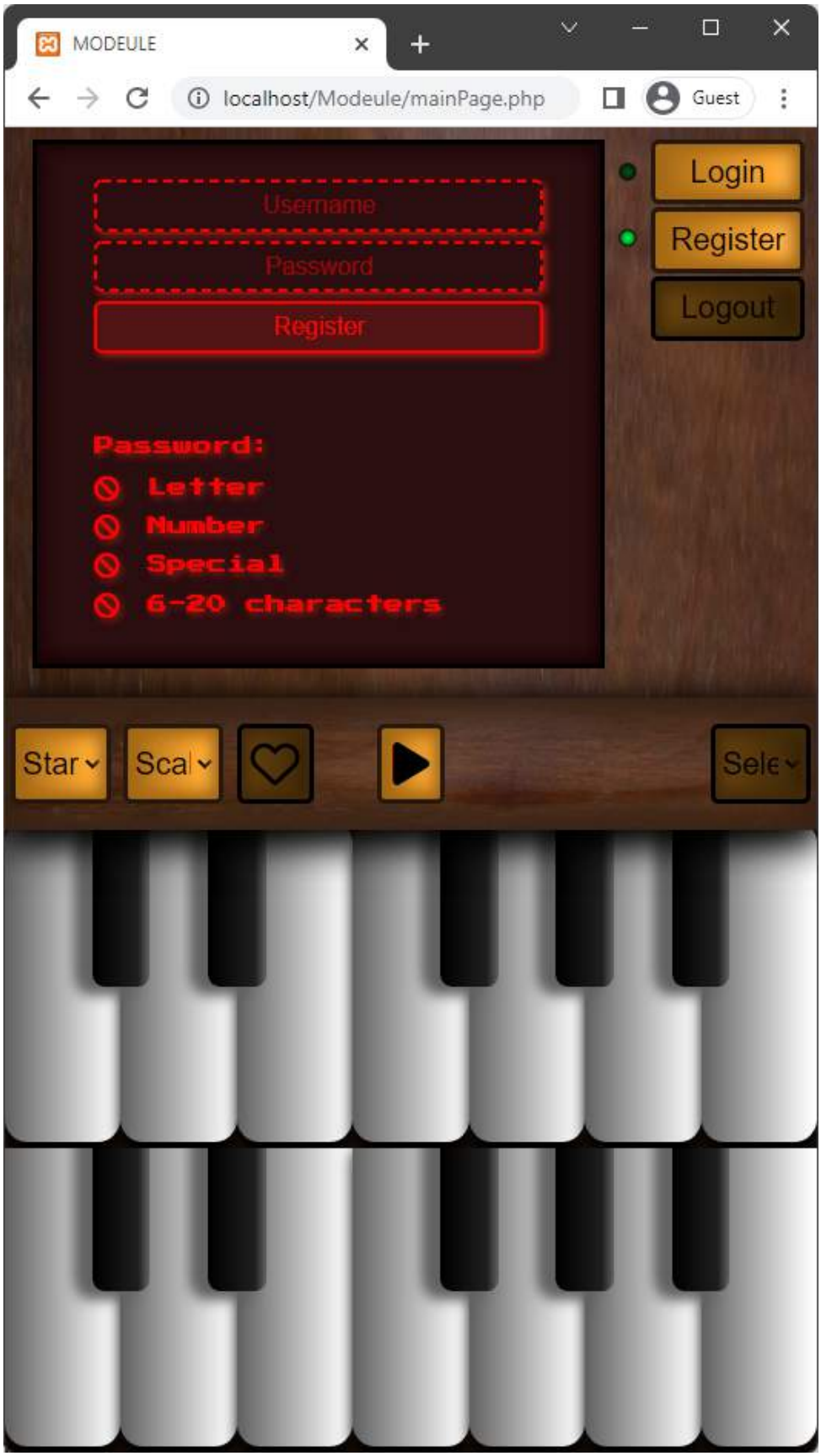
When ‘Start Note’ and ‘Scale Type’ are filled in with values ‘C’ and ‘Ionian’ respectively:



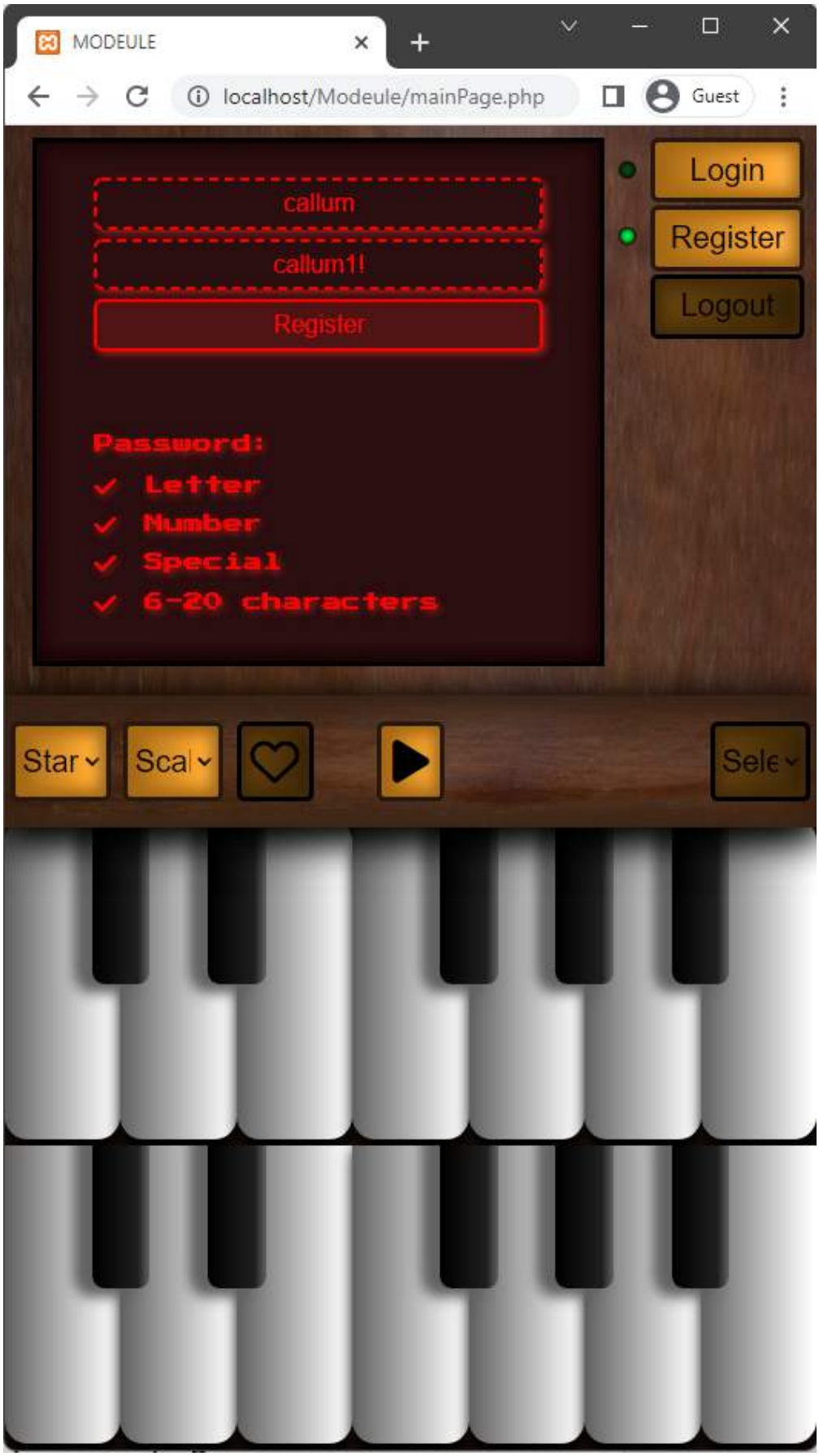
When “Login” button is clicked:



When “Register” button is clicked:

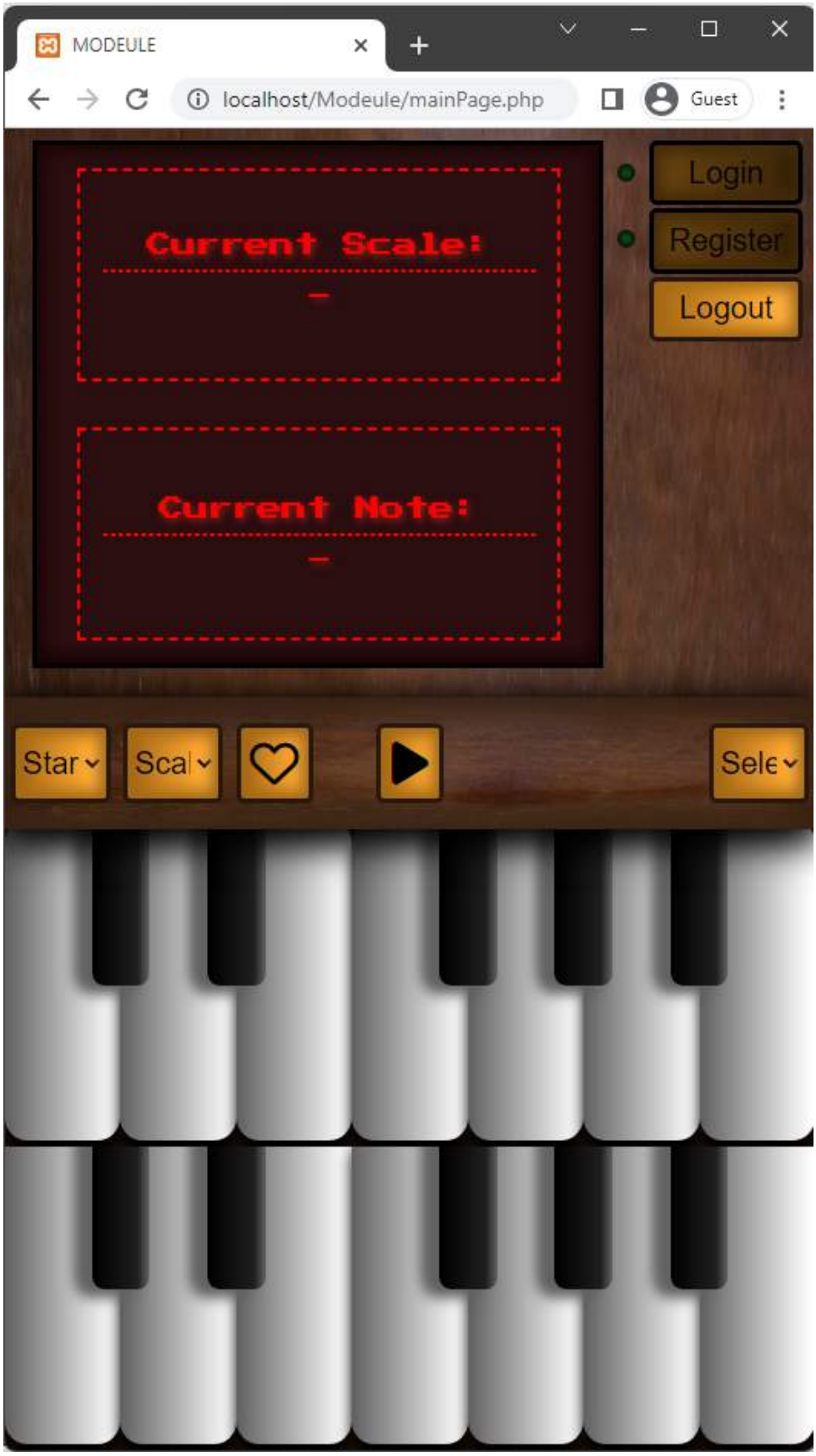


When “Register” button is clicked, and a valid username and password have been entered:

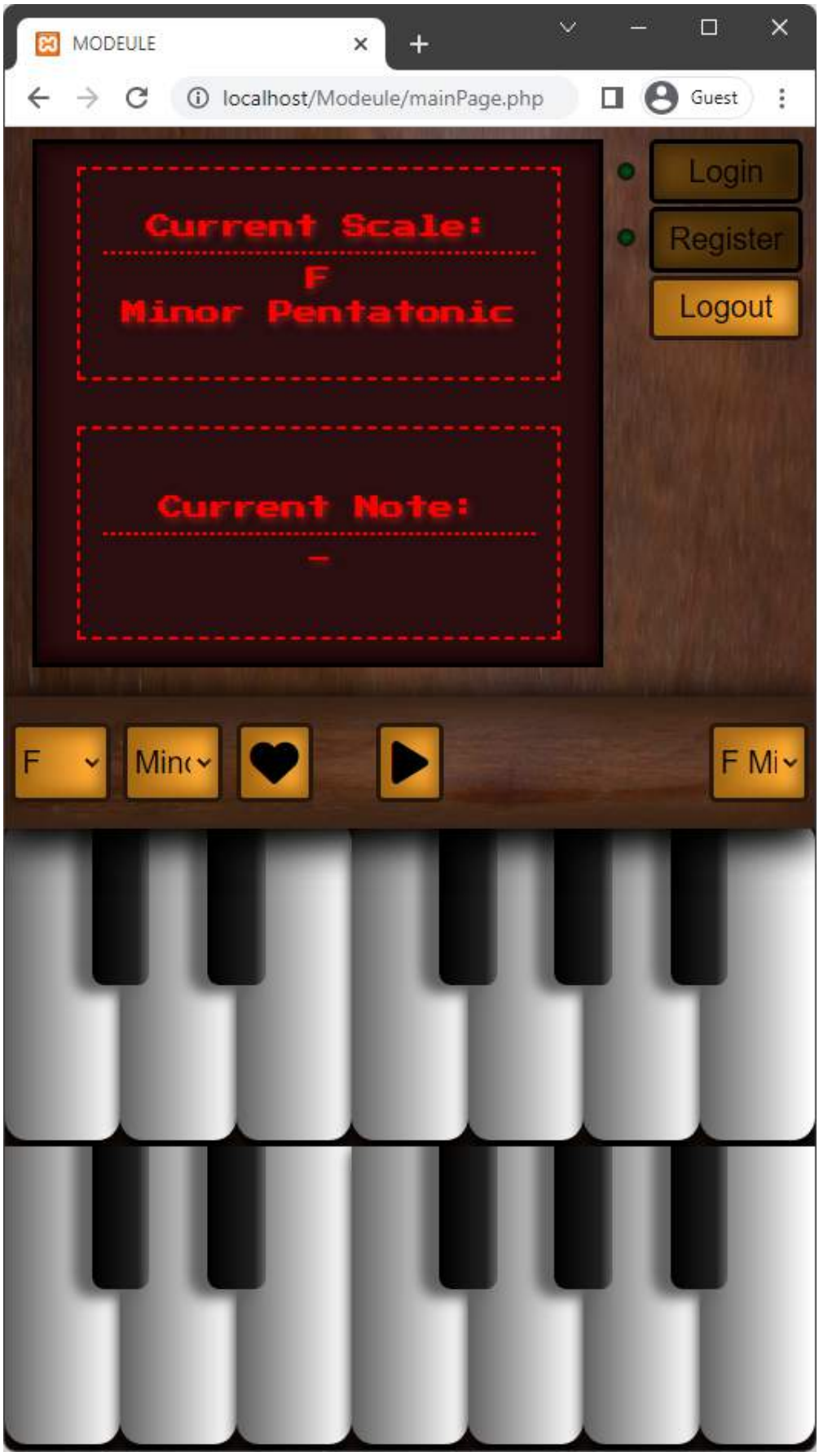


Password is only visible when mouse hovers over it:

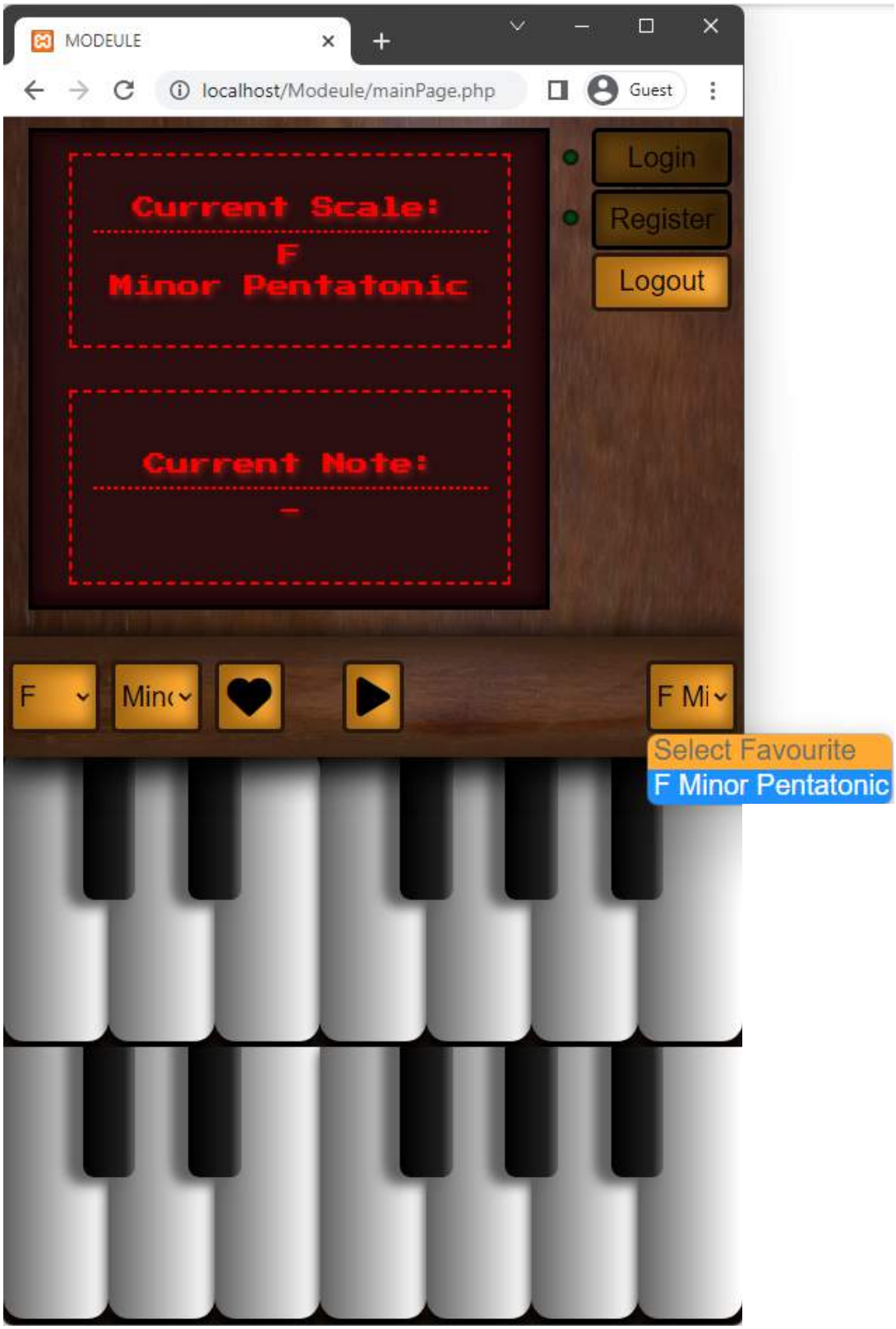
When a user is logged in:



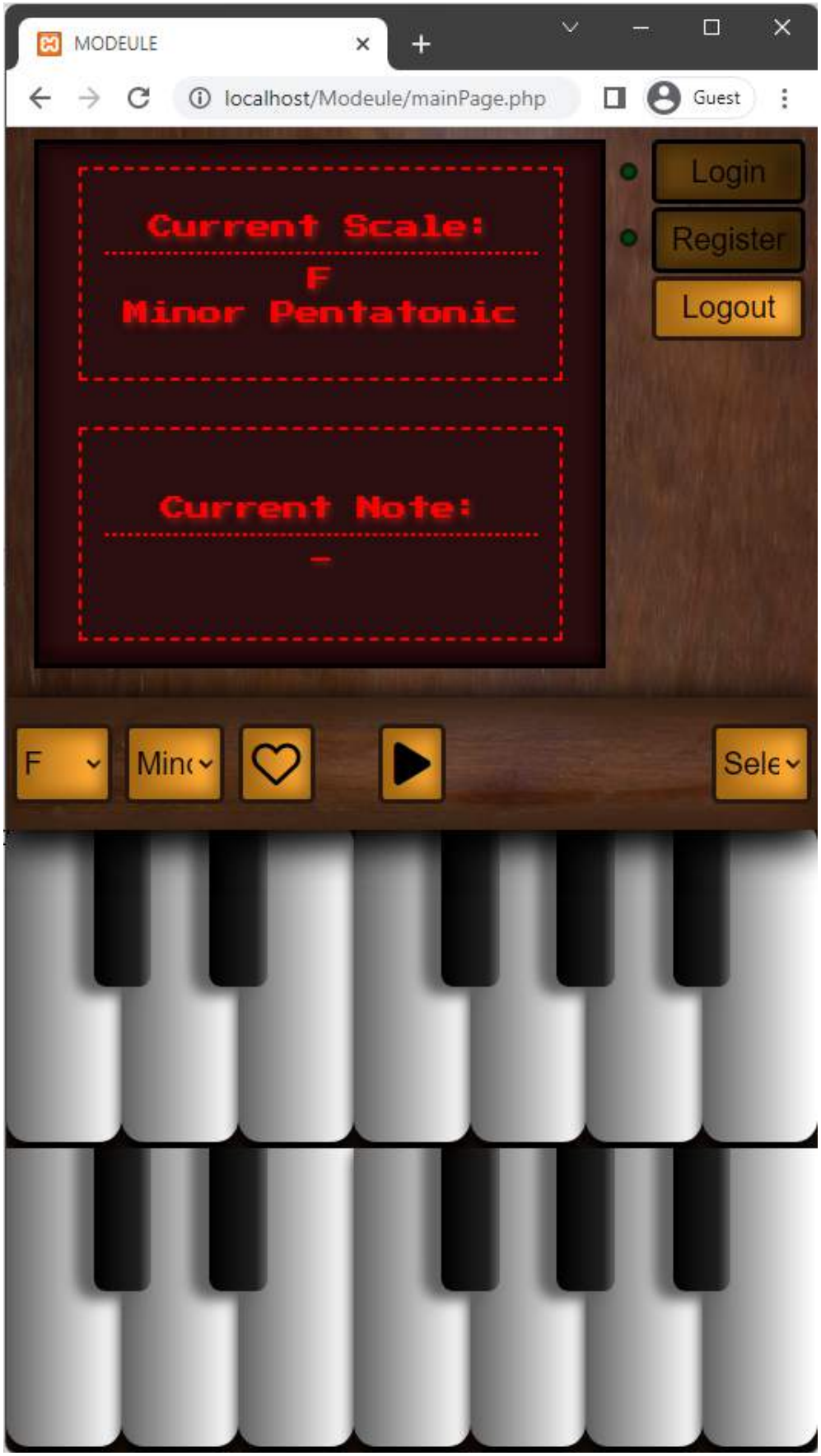
When a logged-in user fills in the scale form, and clicks the 'Favourite' button:



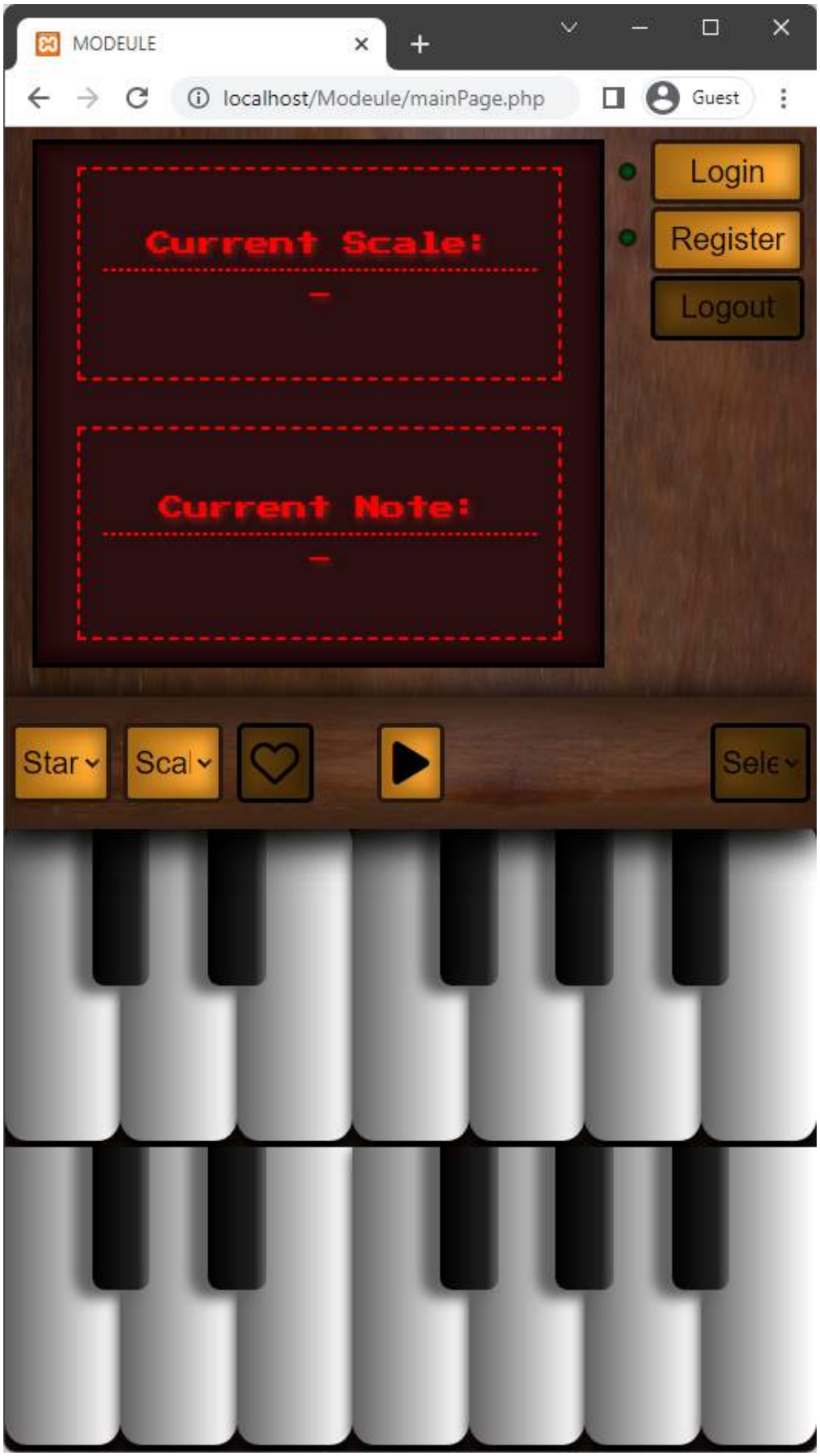
When a logged-in user clicks the ‘Favourites’ dropdown list:



When a logged-in user clicks the ‘Favourite’ button when a favourite is already selected:

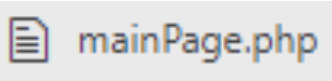


When a logged-in user clicks the ‘Logout’ button:



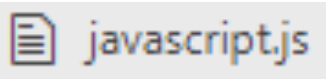
All Files

The files are in a folder named “AH Project”.

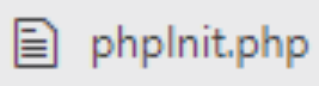


The main page code in HTML. It is a PHP file so the php on the page can run. This code structures the main elements of my page and attaches ID’s and classes to elements for CSS styling

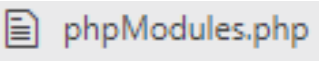
The main styles of the entire website. This includes the default styles, as well as styles that elements can be changed to from JavaScript. At the bottom of this file, some of my media queries can be found.



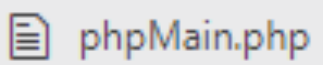
This is my JavaScript file. By far the largest of my files at nearly 500 lines, this file is used for a multitude of things. It creates and spaces out every piano key, in two different styles depending on the width of the window. It styles, disables and toggles the visibility of many elements dependant on form inputs by the user. It is also used to check the password of a registering user.



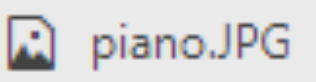
This is a PHP file required at the top of ‘mainPage.php’. It is used to check if a session is to be resumed or restarted. It also connects to the database.



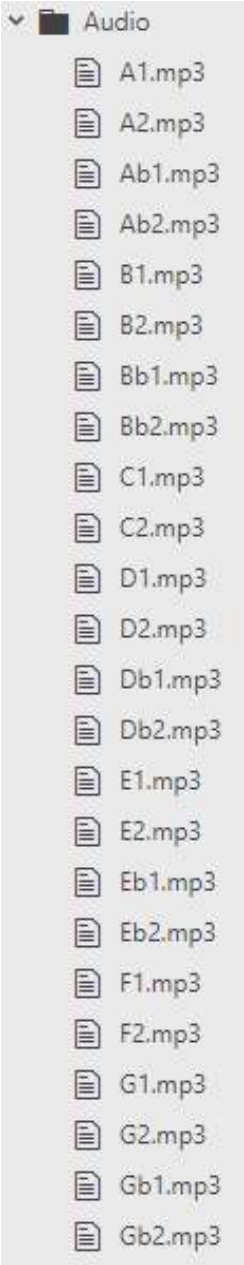
This is a PHP file full of all my high-level PHP code. It contains all the required PHP modules for my code to function. It is essentially a library so that my low-level code can be separate and readable by calling more complicated modules from this file.



My low-level PHP code, a readable file that runs through the core processes. This file executes the main steps by using modules of high-level code from ‘phpModules.php’



This is the picture I used to create a wooden texture for my website. There are no copyright issues because I took this photo myself. To keep the feel of a real piano, this is a close-up picture of my real piano.



This is my folder containing all my sound files. There are no copyright issues because I recorded all of these sounds myself.

HTML Code

mainPage.php

```html
<html>
<head>

<title>Modeules</title>

<!-- Link Javascript File -->
<script type="text/javascript" src="javascript.js"></script>

<!-- -------------------------- -->
<!-- AHC : External CSS Styling -->
<!-- -------------------------- -->

<!-- Link CSS Stylesheet -->
<link rel="stylesheet" href="styles.css">

<!-- Import Icons for forms -->
<script src="https://kit.fontawesome.com/8bcfd9dc90.js" crossorigin="anonymous"></script>

<!-- Import font for screen -->
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Press+Start+2P|Raleway+Dots">

<!-- ------------------------------- -->
<!-- AHC : Open a database connection -->
<!-- ------------------------------- -->

<!-- Connect to database and start PHP session -->
<?php require("phpInit.php"); ?>

</head>

<body>

<!-- Buttons to choose to login or register -->
<div id="loginButtons">

    <!-- ------------------------------------------------ -->
    <!-- AHC : Form Elements using action , method & name -->
    <!-- ------------------------------------------------ -->

    <form action="" method="post">

        <!-- Press to use form to login -->
        <input type="submit" name="loginType" value="Login" id="chooseLogin" class="loginButtons">

        <!-- Press to use form to register -->
        <input type="submit" name="loginType" value="Register" id="chooseRegister" class="loginButtons">

        <!-- Submit button logs current user out. Only visible when logged in -->
        <input type="submit" name="logout" value="Logout" id="chooseLogout" class="loginButtons">

    </form>
</div>
```

```
54 <!-- Create green lights to indicate if the user is logging in or registering -->
55 <div id="selectIndicator">
56     <div class="selectIndicators"><div class="innerSelectIndicators"></div></div>
57     <div class="selectIndicators"><div class="innerSelectIndicators"></div></div>
58 </div>
59
60 <!-- This is the "screen" interface above the keyboard -->
61 <div id="screen">
62
63     <!-- This div splits the screen in two -->
64     <div id="halfScreen">
65
66         <!-- ------------------------------------------------ -->
67         <!-- AHC : Form Elements using action , method & name -->
68         <!-- ------------------------------------------------ -->
69
70         <!-- Form to enter username and password -->
71         <!-- Only visible when logged out -->
72         <!-- Can be used to log in or register depending on which button is pressed -->
73         <div class="loginInput" hidden>
74             <form action="" method="post" autocomplete="off">
75                 <!-- Special data for something ?????? -->
76                 <input type="hidden" name="valid" value="false">
77                 <!-- Enter username -->
78                 <input type="text" name="username" placeholder="Username" minlength="1"
   maxlength="20" required><br>
79                 <!-- Enter password, runs function on key release to validate current password -
   ->
80                <input type="password" name="password" placeholder="Password"
   onkeyup="checkPassword()" onmouseover="showPassword()" onmouseout="hidePassword()" minlength="6"
   maxlength="20" required>
81                 <!-- Sumbit Form -->
82                 <input type="submit" name="enterDetails" value="Login" class="submitButton">
83             </form>
84         </div>
85
86         <!-- Shows which scale has been filled in to the scale form -->
87         <!-- Only visible when not logging in or registering -->
88         <div class="audioInfo" id="scaleTypeInfo">
89             <div>Current Scale:<br><hr><div class="scaleDescription">-</div><div
   class="scaleDescription"></div></div>
90         </div>
91
92     </div>
```

```
94      <div id="halfScreen">
95
96          <!-- Text to help user create a valid password. Only visible when registering -->
97          <!-- Icons update when typing password to indicate if valid -->
98          <div class="loginInput" hidden>
99              <div id="passwordAssist">
100                 <div>Password:</div>
101                 <div>
102                     <i class='fas fa-ban letter'></i>
103                     <i class='fa fa-check letter'></i> Letter
104                 </div>
105                 <div>
106                     <i class='fas fa-ban number'></i>
107                     <i class='fa fa-check number'></i> Number
108                 </div>
109                 <div>
110                     <i class='fas fa-ban special'></i>
111                     <i class='fa fa-check special'></i> Special
112                 </div>
113                 <div>
114                     <i class='fas fa-ban passwordSize'></i>
115                     <i class='fa fa-check passwordSize'></i> 6-20 characters
116                 </div>
117             </div>
118         </div>
119
120         <!-- Shows which scale has been filled in to the scale form -->
121         <!-- Only visible when not logging in or registering -->
122         <div class="audioInfo" id="currentNote">
123             <div>Current Note:<br><hr><div class="scaleDescription">-</div></div>
124         </div>
125
126     </div>
127
128 </div>
129
130 <!-- Image underneath keyboard -->
131 <img src="piano.jpg" id="pianoImage"></img>
```

```
133 <!-- Bar above keyboard -->
134 <div id="pianoImage">
135
136     <!-- First block of forms - alligned to the left -->
137     <div class="formStart">
138
139         <div class="formElements">
140
141             <!-- ------------------------------------------------ -->
142             <!-- AHC : Form Elements using action , method & name -->
143             <!-- ------------------------------------------------ -->
144
145             <!-- Form to choose a scale to play or favourite -->
146             <form action="" method="post" class="keyboardUI" id="scaleForm">
147
148             <!-- Select the note that the scale will start on -->
149             <!-- Each note has a value to offset what note to start on by a certain amount -->
150             <!-- If option is changed, runs checkFavourites() to update the favourite icon -->
151             <select name="startNote" id="startNote" onChange="checkFavourites()"
    class="keyboardUI">
152                 <option value="100" disabled="disabled" selected="selected">Start Note</option>
153                 <option value="0">C</option>
154                 <option value="1">C# / Db</option>
155                 <option value="2">D</option>
156                 <option value="3">D# / Eb</option>
157                 <option value="4">E</option>
158                 <option value="5">F</option>
159                 <option value="6">F# / Gb</option>
160                 <option value="7">G</option>
161                 <option value="8">G# / Ab</option>
162                 <option value="9">A</option>
163                 <option value="10">A# / Bb</option>
164                 <option value="11">B</option>
165             </select>
166
167         </div>
168
169         <div class="formElements">
170
171             <!-- Select the scale that will be played -->
172             <!-- Each scale has a value corresponding to an item in an array of scales -->
173             <!-- If option is changed, runs checkFavourites() to update the favourite icon -->
174             <select name="scaleType" id="scaleType" onChange="checkFavourites()"
    class="keyboardUI">
175                 <option value="100" disabled="disabled" selected="selected">Scale Type</option>
176                 <option value="0">Ionian</option>
177                 <option value="1">Dorian</option>
178                 <option value="2">Phrygian</option>
179                 <option value="3">Lydian</option>
180                 <option value="4">Mixolydian</option>
181                 <option value="5">Aeolian</option>
182                 <option value="6">Locrian</option>
183                 <option value="7">Major Pentatonic</option>
184                 <option value="8">Minor Pentatonic</option>
185                 <option value="9">Major Arpeggio</option>
186                 <option value="10">Minor Arpeggio</option>
187             </select>
188
189             </form>
190
191         </div>
```

```
193         <div class="formElements button favouriteButton">
194             <!-- Favourite Button -->
195             <!-- Submits the start note and scale data to be favourited -->
196             <button type="submit" name="favourite" form="scaleForm" id="favourite"
    class="iconButton">
197             <i class='far fa-heart' id="favouriteIcon"></i>
198             <i class='fas fa-heart' id="unfavouriteIcon"></i>
199             </button>
200         </div>
201
202     </div>
203
204     <div class="formElements button playButton">
205         <!-- Play Button -->
206         <!-- Submits the start note and scale data to be played -->
207         <button type="submit" name="scale" form="scaleForm" class="iconButton"><i class='fas fa-
    play'></i></button>
208     </div>
209
210     <div class="formEnd">
211
212         <div class="formElements">
213
214             <!-- ------------------------------------------------ -->
215             <!-- AHC : Form Elements using action , method & name -->
216             <!-- ------------------------------------------------ -->
217
218             <!-- Form to chose a user-specific favourited scale -->
219             <!-- Fills in start note and scale type for the user when an option is selected -->
220             <form>
221                 <select name="favouritesList" onChange="selectFavourite()" class="keyboardUI"
    id="favouriteForm">
222                     <!-- Default placeholder -->
223                     <option name="favDefault" value="100 100" disabled="disabled"
    selected="selected">Select Favourite</option>
224                 </select>
225             </form>
226         </div>
227
228     </div>
229
230 </div>
231
232 </body>
233
234 <!-- ---------------------------------- -->
235 <!-- AHC : Process form data using PHP -->
236 <!-- ---------------------------------- -->
237
238 <?php
239 require("phpModules.php");
240 require("phpMain.php");
241 ?>
242
243 </html>
```

CSS Code

styles.css

```
/* -------------------------  */
/* AHC : External CSS Styling  */
/* -------------------------  */

/* Grouping selectors, not specific to any one element */

.pianoBackground,.selectIndicators,.loginButtons,#pianoImage,select,.iconButton,#screen input,hr {
  width:100%;
}

body,.pianoBackground {
  background-image:url("piano.jpg");
}

.pianoBackground,#pianoImage {
  left:0;
  box-shadow: 0px 15px 20px #382214 inset, 0px -15px 20px #382214 inset, 0px 5px 25px black;
}

.pianoBackground,.whiteKey,.blackKey,input,select,button,.formElements,.button {
  overflow:hidden;
}

.pianoBackground,.whiteKey,.blackKey,#pianoImage {
  position:absolute;
}

.selectIndicators,#pianoImage,#screen,#halfScreen,.audioInfo {
  display:flex;
  align-items:center;
  justify-content:center;
}

input[type="submit"],select,option,button,.whiteKey,.blackKey {
  cursor:pointer;
}

input:disabled,select:disabled,button:disabled,i,#favouriteForm {
  color:black;
}

.keyboardUI,.iconButton,.formStart,.formEnd,#halfScreen {
  height:100%;
}

i,.audioInfo {
  font-size:200%;
}

#screen input,#passwordAssist,::placeholder,.audioInfo,#screen i {
  color:red;
}

input,select,button,#screen i {
  font-size:125%;
}

#screen input,.audioInfo {
  border:2px dashed red;
}

#halfScreen,.formStart,.formEnd {
  width:50%;
}

#screen input,.audioInfo {
  text-align:center;
}
```

```
70 /* Style body */
71 body {
72   background-size: 100% 100%;
73   background-repeat:no-repeat;
74   z-index:2;
75   cursor:default;
76 }
77
78 .pianoBackground {
79   padding:0;
80   box-shadow: 0px -15px 20px #24160c inset;
81   transition: height 1.5s , bottom 1.5s;
82   /* Get height and displacement from bottom of page */
83   /* Set from javascript */
84   height:var(--height);
85   bottom:var(--fromBottom);
86 }
87
88 .whiteKey,.blackKey {
89   margin-top:0.5%;
90   margin-bottom:0.5%;
91   border-radius:20%/7%;
92   transition-duration:0.2s;
93   right:0;
94   /* Get width and indent to the left */
95   /* Set from javascript */
96   width:var(--keyWidth);
97   left:var(--indentLeft);
98 }
99
100 .whiteKey {
101   background-image: linear-gradient(to right,#636363 2%,#e6e6e6 80%,white 98%);
102   height:97%;
103   box-shadow: 0px 10px 10px black;
104   z-index:0;
105 }
106
107 .whiteKey.pressed {
108   box-shadow: 0px 10px 10px black, -4px 5px 10px #212121 inset, 1px 5px 5px #212121 inset, -1px
    5px 9999px rgba(0,0,0,0.3) inset;
109 }
110
111 .blackKey {
112   background-image: linear-gradient(to right,black 2%,#1f1f1f 80%,#5c5c5c 98%);
113   box-shadow: -10px 5px 10px #636363, -10px 0px 10px #636363;
114   height:50%;
115   z-index:1;
116 }
117
118 .blackKey.pressed {
119   box-shadow: -5px 0px 20px #636363, -5px 0px 5px #636363, -1px 5px 9999px rgba(0,0,0,0.7)
    inset;
120 }
121
122 #selectIndicator,#loginButtons {
123   float:right;
124   height:25%;
125 }
126
127 #loginButtons {
128   width:100px;
129 }
130
131 #selectIndicator {
132   width:30px;
133 }
134
135 .selectIndicators,.loginButtons {
136   height:20%;
137   margin-bottom:2%;
138 }
```

```
140 .innerSelectIndicators{
141   background-image: radial-gradient(#07521c 2%,#053011 50%,#000000 80%);
142   border-radius:100%;
143   width:10px;
144   height:10px;
145   border: 1px solid #26180c;
146 }
147
148 .innerSelectIndicators.lightUp {
149   background-image:radial-gradient(#00ff48 2%,#03a631 50%,#000000 80%);
150 }
151
152 #pianoImage {
153   height:10%;
154   bottom:47%;
155   z-index:4;
156 }
157
158 input,select,button {
159   border-radius: 5px;
160   background-image: linear-gradient(to right,#a66a16 2%,#ffaa33 80%,#ffb957 98%);
161   box-shadow: 5px 5px 10px #a66a16 inset, -5px -5px 10px #a66a16 inset;
162   border: 3px solid #26180c;
163   outline: none;
164 }
165
166 input:disabled,select:disabled,button:disabled {
167   opacity:1;
168   background-image: linear-gradient(to right,#694410 2%,#452c0a 80%,#291a06 98%);
169   box-shadow: 5px 5px 10px #452c0a inset, -5px -5px 10px #452c0a inset;
170   border: 3px solid #000000;
171   cursor:not-allowed;
172 }
173
174 option {
175   background-color:#ffaa33;
176 }
177
178 .formElements {
179   height:60%;
180   margin:2%;
181   width:30%;
182 }
183
184 .formStart,.formEnd {
185   display:flex;
186   align-items:center;
187 }
188
189 .formStart {
190   justify-content:flex-start;
191 }
192
193 .formEnd {
194   justify-content:flex-end;
195 }
196
197 .button {
198   width:50px;
199 }
200
201 .iconButton {
202   font-size:100%;
203 }
204
205 .playButton {
206   width:55px;
207   margin:5%;
208 }
```

```
#screen {
  background-color:#2b0e10;
  width:auto;
  height:40%;
  margin-left:2%;
  display:absolute;
  border: 3px solid black;
  box-shadow: 5px 5px 10px #1c0607 inset, -5px -5px 10px #1c0607 inset;
  font-family: "Press Start 2P", sans-serif;
  text-shadow: 2px 2px 5px #a4231d;
}

#screen input {
  background:transparent;
  box-shadow:none;
  height:15%;
  margin-bottom:5%;
  box-shadow: 2px 2px 5px #a4231d;
}

#screen input[type="submit"] {
  background:rgba(164,35,29,0.3);
  margin-bottom:0;
  border:2px solid red;
}

::placeholder {
  opacity:0.5;
}

.loginInput {
  width:80%;
}

.audioInfo {
  width:85%;
  height:80%;
}

#currentNote {
  margin-right:5%;
}

#scaleTypeInfo {
  margin-left:5%;
}

#scaleTypeInfo div div {
  margin-bottom:2%;
}

.audioInfo > div {
  width:90%;
}

hr {
  height:0px;
  border: 1px dashed red;
}

#screen i {
  margin-right:2%;
}

#passwordAssist > div {
  height: 15%;
}

#unfavouriteIcon {
  display:none;
}
```

```
282 /* ----------------------------------------- */
283 /* AHC : multiple layouts using media queries */
284 /* ----------------------------------------- */
285
286 /* Change screen interface to stack vertically on a smaller screen */
287 @media only screen and (max-width:1000px) {
288
289   #halfScreen,#screen input {
290     width:100%;
291   }
292
293   .audioInfo,#screen input {
294     font-size: 100%;
295   }
296
297   #screen {
298     flex-direction:column;
299   }
300
301   #halfScreen {
302     height:50%;
303   }
304
305   #currentNote {
306     margin-right:0%;
307   }
308
309   #scaleTypeInfo {
310     margin-left:0%;
311   }
312
313   #screen input {
314     height:20%;
315     margin-bottom:0%;
316     margin-top:2%;
317   }
318
319   #passwordAssist > div {
320     font-size:80%;
321   }
322
323 }
```

JavaScript Code

javascript.js

```javascript
// Create a key
// The "keyClass" is either "white" or "black", determining how it is styled
function createKeys(key,keyboard,keyClass,keyWidth,keyIndent) {
  key = document.createElement("div");
  key.classList.add(keyClass);
  key.setAttribute("onclick","keyPress(this)");
  keyboard.appendChild(key);
  key.style.setProperty('--keyWidth',String(keyWidth).concat("%"));
  key.style.setProperty('--indentLeft',String(keyIndent).concat("%"));
}

// Function used to delete all pre-existing keys
function destroyKeys() {
  var keyboardContainer = document.querySelectorAll(".pianoBackground");
  for (i = 0; i < keyboardContainer.length; i++) {
    keyboardContainer[i].remove();
  }
}

// Adds classes to appropriate Divs to style them like keys
// Adds an indent to Divs to space keys appropriately
function buildKeyboard(octaves) {

  // Calculate the number of keys and key widths based on the number of octaves
  var noKeys = octaves*7
  var keyWidthWhite = 100/7/octaves;
  var keyWidthBlack = keyWidthWhite/2;

  // Get all keyboard containers
  var keyboard = document.querySelectorAll(".pianoBackground");

  // Loop for the number of keyboards
  for (j = 0; j < keyboard.length; j++) {

    // Initialise the indent of white and black keys
    var indentWhite = 0;
    var indentBlack = keyWidthBlack*1.5;

    // Create array to store all piano key divs
    var keys = new Array(noKeys);

    // Loop for every key
    for (i = 0; i < noKeys; i++) {

      // Create a white key
      createKeys(keys[i],keyboard[j],"whiteKey",keyWidthWhite,indentWhite);

      // Create a black key depending on the counter
      if (i/2 == 1 || i/6 == 1 || i/9 == 1 || i/13 == 1) {}else{
        createKeys(keys[i],keyboard[j],"blackKey",keyWidthBlack,indentBlack);
      }

      // Increase the indent of the next key
      indentWhite += keyWidthWhite;
      indentBlack += 2*keyWidthBlack;
    }
  }
}
```

```
60 // Create Div Elements required for the narrow window piano
61 function createKeysNarrow(init) {
62
63   // Remove all pre-existing key elements
64   destroyKeys();
65   var body = document.getElementsByTagName("body")[0];
66
67   // Create array to store two elements to contain a keyboard each
68   var keyboardContainer = new Array(2);
69
70   // Create both keyboard containers
71   for (i = 0; i < 2; i++) {
72     keyboardContainer[i] = document.createElement("div");
73     keyboardContainer[i].classList.add("pianoBackground");
74     body.appendChild(keyboardContainer[i]);
75     keyboardContainer[i].style.setProperty("--fromBottom","0");
76   }
77
78   // Bring top keyboard above the bottom one and set height of bottom keyboard to 25%
79   keyboardContainer[1].style.zIndex="3";
80   keyboardContainer[0].style.setProperty("--height","25%");
81
82   // Function to set final styles of top keyboard (25% height & displaced upwards by 23%)
83   function narrowKeyStyles() {
84       keyboardContainer[1].style.setProperty("--height","25%");
85       keyboardContainer[1].style.setProperty("--fromBottom","23%");
86   }
87
88   // Check if animation is required. Only required if triggered by media query.
89   // Build keyboard in both cases.
90   // Pass in 1 to indicate that only 1 octave needs to be built in each div
91   if (init == false) {
92
93     // Triggered by media query so animate.
94     keyboardContainer[1].style.setProperty("--height","50%");
95     buildKeyboard(1);
96     setTimeout(narrowKeyStyles,1);
97   }else{
98
99     // Do not animate. Triggered by a refresh or a submit.
100    narrowKeyStyles(keyboardContainer);
101    buildKeyboard(1);
102  }
103 }
```

```
105 // Create Div Elements required for a wide window piano
106 function createKeysWide(init) {
107
108   // Remove all pre-existing key elements
109   destroyKeys();
110   var body = document.getElementsByTagName("BODY")[0];
111
112   // Create variable store an elements to contain the keyboard
113   keyboardContainer = document.createElement("div");
114   keyboardContainer.classList.add("pianoBackground");
115   body.appendChild(keyboardContainer);
116   keyboardContainer.style.setProperty("--height","25%");
117   keyboardContainer.style.setProperty("--fromBottom","23%");
118
119   // Function to set final styles of the keyboard (50% height & at bottom of screen)
120   function wideKeyStyles() {
121     keyboardContainer.style.setProperty("--height","50%");
122     keyboardContainer.style.setProperty("--fromBottom","0");
123   }
124
125   // Check if animation is required. Only required if triggered by media query.
126   // Build keyboard in both cases.
127   // Pass in 2 to indicate that 2 octaves need to be built
128   if (init == false) {
129     buildKeyboard(2);
130     setTimeout(wideKeyStyles,1);
131   }else{
132     wideKeyStyles();
133     buildKeyboard(2);
134   }
135 }
136
137 // ----------------------------------------
138 // AHC : multiple layouts using media queries
139 // ----------------------------------------
140
141 // Create event listener to check the window width
142 // If the width is changed, update piano interface
143 var resizeCheck = window.matchMedia("(max-width: 720px)");
144 window.addEventListener("resize",mediaQuery);
145
146 // Check the screen width event listener.
147 // Construct a piano layout depending on screen width
148 function mediaQuery(init) {
149   if (init != true) {init = false;}
150   globalThis.screenConstant;
151   screenIsSmall = resizeCheck.matches;
152   if (screenConstant != screenIsSmall) {
153     screenConstant = screenIsSmall;
154     if (screenIsSmall == false) {
155       createKeysWide(init);
156     }else{
157       createKeysNarrow(init);
158     }
159   }
160 }
161
162 // Initial construction of piano keyboard.
163 function mediaQueryInit() {
164   globalThis.screenConstant = !resizeCheck.matches;
165   mediaQuery(true);
166 }
```

```
168 // Array to store names of all sound files
169 var allSounds =
    ["C1","Db1","D1","Eb1","E1","F1","Gb1","G1","Ab1","A1","Bb1","B1","C2","Db2","D2","Eb2","E2","F2
    ","Gb2","G2","Ab2","A2","Bb2","B2"];
170 // Array of note names and scale types. Used to construct the name of a scale for options in the
    "favourite" select input.
171 var notes = ["C","C#/Db","D","D#/Eb","E","F","F#/Gb","G","G#/Ab","A","A#/Bb","B"];
172 var scales = ["Ionian","Dorian","Phrygian","Lydian","Mixolydian","Aeolian","Locrian","Major
    Pentatonic","Minor Pentatonic","Major Arpeggio","Minor Arpeggio"];
173
174 // Function to play a key
175 // Runs when a key is pressed, also used to play each note of a scale
176 function keyPress(key) {
177
178   // Get all keys into array
179   var allKeys = document.querySelectorAll(".whiteKey,.blackKey");
180
181   // Start animation on the key and create audio element
182   key.classList.add("pressed");
183   audioPlay = document.createElement("audio");
184
185   // Set timer to set the key to normal and delete the audio element after 0.5s
186   setTimeout(function time(){
187     key.classList.remove("pressed");
188     audioPlay.remove();
189   }, 500);
190
191   // Loop for every key
192   for (i = 0; i < allKeys.length; i++) {
193
194     // If key matches the key that was clicked
195     if (key == allKeys[i]) {
196
197       // Add the correct source to the audio tag and play
198       var source = "Audio//".concat(globalThis.allSounds[i].concat(".mp3"))
199       audioPlay.src = source;
200       audioPlay.play();
201
202       // Update the screen to display which note is being played
203       var j = i;
204       if (j > 11) {
205         j -= 12;
206       }
207       showInfo();
208       document.querySelectorAll(".scaleDescription")[2].innerHTML=notes[j];
209       break;
210
211     }
212   }
213 }
```

```
// Scale types, all stored in arrays.
// Each item represents a position in an octave.
// The scale will be played up and down.
var ionian = [0,2,4,5,7,9,11,12];
var dorian = [0,2,3,5,7,9,10,12];
var phrygian = [0,1,3,5,7,8,10,12];
var lydian = [0,2,4,6,7,9,11,12];
var mixolydian = [0,2,4,5,7,9,10,12];
var aeolian = [0,2,3,5,7,8,10,12];
var lochrian = [0,1,3,5,6,8,10,12];
var majPent = [0,2,4,7,9,12];
var minPent = [0,3,5,7,10,12];
var majArp = [0,4,7,12];
var minArp = [0,3,7,12];

// Insert all scales into one long array, corresponding to the value of the "scaleType" select input.
var allScales = [ionian,dorian,phrygian,lydian,mixolydian,aeolian,lochrian,majPent,minPent,majArp,minArp];

// Function to play a selected scale. Correct start note and scale type is passsed in.
// Uses the keyPress() function to play each key
function playScale(startNote,scaleType) {

  // Get correct scale and format to include the rest of the scale in reverse.
  var scale = allScales[scaleType];
  var scaleReverse =  scale.map((x) => x);scaleReverse.pop();scaleReverse.reverse();
  var finalScale = scale.concat(scaleReverse);

  // Get all keys into an array
  var allKeys = document.querySelectorAll(".whiteKey,.blackKey");
  var j = 0;

  // Loop every 0.5s to play each note of the scale until complete
  var metronome = setInterval(function time() {

    // Offset scale by startNote
    var keyPos = finalScale[j] + startNote;
    var key = allKeys[keyPos];
    keyPress(key);
    j += 1;
    if (j == finalScale.length){
      clearInterval(metronome);
    }

  }, 500);

}

// If the type of login has changed, save to localStorage for use after the page refreshes
function loginTypeChange() {
  localStorage.setItem("loginTypeChange","true");
}

// Function used to switch off the indicator lights next to the login type buttons
function indicatorLightsOff() {
  var indicatorLights = document.querySelectorAll(".innerSelectIndicators");
  indicatorLights[0].classList.remove("lightUp");
  indicatorLights[1].classList.remove("lightUp");
}
```

```
// Update the screen to show either the scale information or the login form
// Also update the indicator lights next to the login type buttons
function showInfo() {
  var state = localStorage.getItem("loginTypeChange");
  for (i = 0; i < 2; i++) {
    if (state == "true") {
      document.querySelectorAll(".audioInfo")[i].style.display="none";
      document.querySelectorAll(".loginInput")[i].hidden=false;
      if (localStorage.getItem("loginType") == "Login") {
        document.querySelectorAll("#halfScreen")[1].style.visibility="hidden";
      }else{
        document.querySelectorAll("#halfScreen")[1].style.visibility="visible";
      }
    }else{
      indicatorLightsOff();
      document.querySelectorAll(".audioInfo")[i].style.display="flex";
      document.querySelectorAll(".loginInput")[i].hidden=true;
      document.querySelectorAll("#halfScreen")[1].style.visibility="visible";
    }
  }
  localStorage.setItem("loginTypeChange",false);
}

// Update password assistance on the screen when registering
function updatePasswordAssist(password,type,assistClass) {
  if (type.test(password)) {
    document.querySelectorAll(assistClass)[0].style.display="none";
    document.querySelectorAll(assistClass)[1].style.display="inline";
  }else{
    document.querySelectorAll(assistClass)[0].style.display="inline";
    document.querySelectorAll(assistClass)[1].style.display="none";
  }
}
```

```
// Validate Password - ran every time a user enters a character into the password input
function checkPassword() {

  // Only validate if the user is loggin in
  if (localStorage.getItem("loginType") == "Register") {

    // Check for special character, number and letter
    var password = document.getElementsByName("password")[0].value;
    var special = /[!@#$%^&*()_+\-=\[\]{};':"\\|,.<>\/?]+/;
    var letter = /[a-zA-Z]/;
    var number = /\d/;

    // Update value of hidden form element. This is used in php to check if the password is valid or not
    if (special.test(password) && letter.test(password) && number.test(password)) {
      document.getElementsByName("valid")[0].value = "true";
    }else{
      document.getElementsByName("valid")[0].value = "false";
    }

    // Update screen depending on how valid the password is
    updatePasswordAssist(password,letter,".letter");
    updatePasswordAssist(password,number,".number");
    updatePasswordAssist(password,special,".special");
    if (password.length >= 6 && password.length <= 20) {
      document.querySelectorAll(".passwordSize")[0].style.display="none";
      document.querySelectorAll(".passwordSize")[1].style.display="inline";
    }else{
      document.querySelectorAll(".passwordSize")[0].style.display="inline";
      document.querySelectorAll(".passwordSize")[1].style.display="none";
    }
  }
}

// Update value of login submit to inform the user if they are loggin in or registering
// Save the login type to local storage to use elsewhere
function updateLoginType(loginType) {
    document.querySelector(".submitButton").value=loginType;
  localStorage.setItem("loginType",loginType);
  var indicatorLights = document.querySelectorAll(".innerSelectIndicators");
  if (loginType == "Login") {
    indicatorLights[0].classList.add("lightUp");
    indicatorLights[1].classList.remove("lightUp");
  }else{
    indicatorLights[1].classList.add("lightUp");
    indicatorLights[0].classList.remove("lightUp");
  }
  // Update password validation
  checkPassword();
}
```

```
358 // Save the scale form so that it does not reset if the page is refreshed
359 function keepScale(startNote,scaleType) {
360
361   // Update the screen above the keyboard
362   showInfo();
363
364   // If scale is favourited, indicate in favourite select element
365   var scaleInfo = String(startNote).concat(" ").concat(String(scaleType));
366   var favouriteOption = document.querySelectorAll(".favouriteOption");
367   for (i = 0; i < favouriteOption.length; i++) {
368     if (favouriteOption[i].value == scaleInfo) {
369       document.getElementsByName("favouritesList")[0].value=scaleInfo;
370     }
371   }
372
373   // Get the start note and scale type
374   document.getElementById("startNote").value=startNote;
375   document.getElementById("scaleType").value=scaleType;
376   if (startNote != "100" && scaleType != "100") {
377     var screenScaleNote = document.querySelectorAll(".scaleDescription")[0];
378     var screenScaleType = document.querySelectorAll(".scaleDescription")[1];
379
380     // display scale type on screen
381     screenScaleNote.innerHTML=notes[startNote];
382     screenScaleType.innerHTML=scales[scaleType];
383   }
384   document.querySelectorAll(".scaleDescription")[2].innerHTML="-";
385 }
386
387 // Save the username & password so that they does not reset if the page is refreshed
388 function keepForms(username,password) {
389   if (username != "") {
390     document.getElementsByName("username")[0].value=username;
391   }
392   if (password != "") {
393     document.getElementsByName("password")[0].value=password;
394   }
395 }
396
397 // Disable elements depending on if the user is logged in or not
398 function disableInput(id,state) {
399   if (state == "true") {
400     document.getElementById(id).disabled=true;
401   }else{
402     document.getElementById(id).disabled=false;
403   }
404 }
405
406 // When user hovers cursor over password input box, make the password visible
407 function showPassword() {
408   document.getElementsByName("password")[0].type="text";
409 }
410
411 // When user's cursor is not over the password input box
412 function hidePassword() {
413   document.getElementsByName("password")[0].type="password";
414 }
```

```
416 // Delete all options in favourite select
417 // Create new options taken directly from favourites table
418 function updateFavourites(startNote,scaleType) {
419   var scaleString = notes[startNote].concat(" ").concat(scales[scaleType]);
420   var scaleInfo = String(startNote).concat(" ").concat(String(scaleType));
421   var favouriteOption = document.querySelectorAll(".favouriteOption");
422   favouriteOption = document.createElement("option");
423   favouriteOption.classList.add("favouriteOption");
424   favouriteOption.innerHTML = scaleString;
425   favouriteOption.value = scaleInfo;
426   favouriteOption.selected = "selected";
427   document.getElementsByName("favouritesList")[0].appendChild(favouriteOption);
428 }
429
430 // Update favourite icon to hollow or filled
431 function updateFavouriteIcon(hollow,fill,deleted) {
432   hollow = "display:".concat(hollow);
433   fill = "display:".concat(fill);
434   document.getElementById("favouriteIcon").style=hollow;
435   document.getElementById("unfavouriteIcon").style=fill;
436   if (deleted == "true") {
437     document.getElementsByName("favDefault")[0].selected = "selected";
438   }
439 }
440
441 // If user selects different favourite option, fill in scales form appropriately
442 function selectFavourite() {
443   var values = document.getElementsByName("favouritesList")[0].value;
444   const noteAndScale = values.split(" ");
445   var startNote = noteAndScale[0];
446   var scaleType = noteAndScale[1];
447   keepScale(startNote,scaleType);
448   if (values != "100 100") {
449     updateFavouriteIcon("none","block");
450   }else{
451     updateFavouriteIcon("block","none");
452   }
453 }
454
455 // When user selects a different option from the scales form,  screen and favourite select input
456 function checkFavourites(startNote,scaleType) {
457   showInfo();
458   var favouriteOption = document.querySelectorAll(".favouriteOption");
459   if (!startNote && !scaleType && startNote != 0 && scaleType != 0) {
460     var startNote = document.getElementsByName("startNote")[0].value;
461     var scaleType = document.getElementsByName("scaleType")[0].value;
462   }
463   if (startNote != "100" && scaleType != "100") {
464     // display scale type on screen
465     var screenScaleNote = document.querySelectorAll(".scaleDescription")[0];
466     var screenScaleType = document.querySelectorAll(".scaleDescription")[1];
467     screenScaleNote.innerHTML=notes[startNote];
468     screenScaleType.innerHTML=scales[scaleType];
469   }
470   var scaleInfo = String(startNote).concat(" ").concat(String(scaleType));
471   document.getElementsByName("favDefault")[0].selected = "selected";
472   updateFavouriteIcon("block","none");
473   for (i = 0; i < favouriteOption.length; i++) {
474     if (favouriteOption[i].value == scaleInfo) {
475       favouriteOption[i].selected = "selected";
476       updateFavouriteIcon("none","block");
477     }
478   }
479 }
```

PHP CODE

PHP INIT phpInit.php

```php
<?php

// If there is an active session
if (isset($_SESSION)) {
  // If logout button has been pressed
  if ($_SESSION["logout"] == "true") {

    // ---------------------
    // AHC : Close a session
    // ---------------------

    // Destroy the session
    session_destroy();
  }
}

// ---------------------
// AHC : Start a session
// ---------------------

// Start a new session
session_start();

// ---------------------------
// AHC : Connect to a database
// ---------------------------

// Function to connect to a database
function dbConnect($host,$db_username,$db_password,$database) {
  // Values for host , username , password and database are passed in
  // Attempt to connect to database
  $conn = mysqli_connect("$host","$db_username","$db_password","$database");
  // If unsuccessful, close connection.
  if (!$conn) {
    mysqli_close($conn);
  }

  // ------------------------------
  // AHC : Use of session variables
  // ------------------------------

  // Set connection as session variable
  $_SESSION["conn"] = $conn;

}

// Connect to database with correct data
dbConnect("localhost","160688033","160688033","AH Project");

?>
```

PHP MODULES phpModules.php

```php
<?php

// -------------------------------------------------------
// AHC : Use of session variables - used in most functions
// -------------------------------------------------------

// Function to disable forms depending on if user is logged in
function disableForms($state1,$state2) {
    echo "<script>disableInput('favourite','$state1');</script>";
    echo "<script>disableInput('chooseLogout','$state1');</script>";
    echo "<script>disableInput('favouriteForm','$state1');</script>";
    echo "<script>disableInput('chooseLogin','$state2');</script>";
    echo "<script>disableInput('chooseRegister','$state2');</script>";
}

// Function to efficiently retrieve start note and scaletype
function getScaleInfo() {
    $_SESSION["startNote"] = $_POST["startNote"];
    $_SESSION["scaleType"] = $_POST["scaleType"];
}

// Function to attempt to log the user in
function loginCheck() {

    // Get database connection
    $conn = $_SESSION["conn"];

    // Get the username and password
    $username = $_SESSION["username"];
    $password = $_SESSION["password"];

    // -------------------------------------------------------
    // AHC : Execute SQL query to collect data from a database
    // Q1 : To retrieve the password linked to a username
    // -------------------------------------------------------

    // Run query to check if the password matches an existing username
    $query = 'SELECT password FROM users WHERE username LIKE "%'.$username.'%"';
    $rows = mysqli_query($conn, $query) or die(mysqli_error($conn));

    // ----------------------------------------------
    // AHC : Format and process result of an SQL query
    // ----------------------------------------------

    // If there is nothing returned the username is invalid
    if (mysqli_num_rows($rows)==0) {
      $_SESSION["loggedIn"] = "false";
    }else{
      // If the password doesn't match, user will not be logged in
      if (mysqli_fetch_array($rows)['password'] != $password) {
        $_SESSION["loggedIn"] = "false";
      }else{
        // Login successful
        $_SESSION["loggedIn"] = "true";
      }
    }

}
```

```
60 // Function to attempt to register a new user
61 function registerCheck() {
62
63     // Get database connection
64     $conn = $_SESSION["conn"];
65
66     // Get the username and password
67     $username = $_SESSION["username"];
68     $password = $_SESSION["password"];
69
70     // ---------------------------------------------------------
71     // AHC : Execute SQL query to insert new data into a database
72     // Q2 : To insert a new user into the 'users' table
73     // ---------------------------------------------------------
74
75   // Run query to add new user to the "users" table
76   $query = "INSERT INTO users (username,password) VALUES ('$username','$password')";
77   $queryData = mysqli_query($conn,$query) or die (mysqli_error($conn));
78
79     // Log the new user in
80     loginCheck();
81
82 }
```

```
84 // Add (or remove) a favourite scale
85 function addFavourite() {
86
87     // Get database connection
88     $conn = $_SESSION["conn"];
89
90     // Get current username, the start note and the scale type
91     $username = $_SESSION["username"];
92     $startNote = $_SESSION["startNote"];
93     $scaleType = $_SESSION["scaleType"];
94
95     // If both start note and scale type have been selected
96     if ($startNote != 100 && $scaleType != 100) {
97
98         // ---------------------------------------------------------------
99         // AHC : Execute SQL query to collect data from a database
100        // Q3 : To retrieve all the favourites from the 'favourites' table
101        // ---------------------------------------------------------------
102
103       // Run query to get all favourite scales from the favourites database
104       $query = 'SELECT startNote,scaleType FROM favourites WHERE username LIKE
    "%".$username."%"';
105       $queryData = mysqli_query($conn,$query) or die (mysqli_error($conn));
106
107       // Assume the scale is not favouirited
108       $favourited = "false";
109
110         // ------------------------------------------------
111         // AHC : Format and process result of an SQL query
112         // ------------------------------------------------
113
114       // Check all rows for a scale matching the session data.
115       if (mysqli_num_rows($queryData) > 0) {
116         while ($row = mysqli_fetch_array($queryData)) {
117           if ($row["startNote"] == $startNote && $row["scaleType"] == $scaleType) {
118             // A match has been found so update favourited boolean
119             $favourited = "true";
120           }
121         }
122       }
123
124       if ($favourited == "true") {
125
126             // -------------------------------------------------------
127             // AHC : Execute SQL query to delete data from a database
128             // Q4 : To delete a favourite scale
129             // -------------------------------------------------------
130
131             // If a matching favourite was found, delete it. This means the user has
    unfavourited a scale
132         $query = "DELETE FROM favourites WHERE startNote=".$startNote." AND
    scaleType=".$scaleType." AND username LIKE '%".$username."%'";
133         $queryData = mysqli_query($conn,$query) or die(mysqli_error($conn));
134
135         // Update session so the favourite input reverts to default option
136         $_SESSION["deleted"] = "true";
137
138       }else{
139
140             // ---------------------------------------------------------
141             // AHC : Execute SQL query to insert new data into a database
142             // Q5 : To add a new favourite to the 'favourites' table
143             // ---------------------------------------------------------
144
145             // If there is no favourite that matches, add it to the table
146         $query = "INSERT INTO favourites (username,startNote,scaleType) VALUES
    ('$username','$startNote','$scaleType')";
147         $queryData = mysqli_query($conn,$query) or die(mysqli_error($conn));
148
149         // Set filled state for the favourite button
150         $_SESSION["hollow"] = "none";
151         $_SESSION["fill"] = "block";
152       }
153     }
154
155 }
```

```
157 // Retrive the current user's favourites and update the list
158 function updateFavouriteList() {
159
160     // Get database connection
161     $conn = $_SESSION["conn"];
162
163     // Get the current username
164     $username = $_SESSION["username"];
165
166     // -------------------------------------------------------
167     // AHC : Execute SQL query to collect data from a database
168     // Q6 : To collect all of one user's favourites
169     // -------------------------------------------------------
170
171     // Run query to get all records from the "favourites" table with matching username
172     $query = 'SELECT * FROM favourites WHERE username LIKE "%'.$username.'%"';
173     $queryData = mysqli_query($conn,$query) or die (mysqli_error($conn));
174
175     // ----------------------------------------------
176     // AHC : Format and process result of an SQL query
177     // ----------------------------------------------
178
179     // If a result was found
180     if (mysqli_num_rows($queryData) > 0) {
181       // Loop through all records
182       while ($row = mysqli_fetch_array($queryData)) {
183         // Get the start note and scale type of each favourite
184         $startNote = $row["startNote"];
185         $scaleType = $row["scaleType"];
186         // Create the option in the favourite select input
187         echo "<script>updateFavourites(".$startNote.",".$scaleType.");</script>";
188       }
189     }
190
191 }
```

```
193 // Update whole page
194 function updatePage() {
195
196     // Get database connection
197     $conn = $_SESSION["conn"];
198
199   // If login type is set:
200   if ($_SESSION["loginType"] == "Login" || $_SESSION["loginType"] == "Register") {
201     // If login type is Login:
202     if ($_SESSION["loginType"] == "Login") {
203         // Update in javascript
204         echo "<script>updateLoginType('Login');</script>";
205     // If login type is Register
206     }else{
207         // Update in javascript
208         echo "<script>updateLoginType('Register');</script>";
209         // Update password assistance
210         echo "<script>checkPassword();</script>";
211     }
212   }
213
214     // If the user is logged in:
215     if ($_SESSION["loggedIn"] == "true") {
216         // Update the user's favourites list
217         updateFavouriteList();
218         // Disable appropriate forms (login/register)
219         disableForms("false","true");
220         // Update favourite button
221         echo "
    <script>updateFavouriteIcon('".$_SESSION["hollow"]."','".$_SESSION["fill"]."','".$_SESSION["dele
    ted"]."');</script>";
222         // Make sure scale form is consistent with selected favourite option
223         echo "<script>checkFavourites(".$_SESSION["startNote"].",".$_SESSION["scaleType"].");
    </script>";
224         // Switch off indicator lights
225         echo "<script>indicatorLightsOff()</script>";
226     // If user is not logged in
227     }else{
228         // Disable appropriate forms (favourite,logout)
229         disableForms("true","false");
230     }
231
232     // If a login / register attempt is invalid, forms will be filled in with the attempt data
233     echo "<script>keepForms('".$_SESSION["username"]."','".$_SESSION["password"]."');</script>";
234     // If page has been refreshed, restore scale form to keep values from before the refresh
235     echo "<script>keepScale(".$_SESSION["startNote"].",".$_SESSION["scaleType"].");</script>";
236
237 }
238
239 ?>
```

PHP MAIN phpMain.php

```php
<?php

// Create keyboard when page has loaded
echo "<script>mediaQueryInit();</script>";

// ------------------------------
// AHC : Use of session variables
// ------------------------------

// Check if any forms had been submitted before the page refreshed
$_SESSION["isSubmitted"] = isset($_POST['loginType']) ||
                           isset($_POST['enterDetails']) ||
                           isset($_POST['scale']) ||
                           isset($_POST['favourite']);

// If nothing has been submitted, initialise all important session variables
// Also set indicator lights and appropriate forms to their initial state
if ($_SESSION["isSubmitted"] == 0) {
    $_SESSION["username"] = "";
    $_SESSION["password"] = "";
    $_SESSION["startNote"] = "100";
    $_SESSION["scaleType"] = "100";
  $_SESSION["loggedIn"] = "false";
    $_SESSION["loginType"] = "";
    echo "<script>indicatorLightsOff();</script>";
    disableForms("true","false");
}

// Set initial states of favourite-related variables
// Initial state of "favourite" button
$_SESSION["hollow"] = "block";
$_SESSION["fill"] = "none";
// Initially "deleted" is false (a favourite has NOT been deleted)
$_SESSION["deleted"] = "false";
```

```
36 // -----------------------
37 // AHC : Process form data
38 // -----------------------
39
40 // Check inputs from the user
41 // If any submit buttons have been pressed:
42 if ($_SERVER["REQUEST_METHOD"] == "POST") {
43     if ($_SESSION["isSubmitted"] == 1 || isset($_POST['logout'])) {
44
45         // If login type has been changed:
46         if (isset($_POST['loginType'])) {
47             $_SESSION["loginType"] = $_POST["loginType"];
48             // Update in javascript as well
49             echo "<script>loginTypeChange()</script>";
50         }
51
52         // If login/register button has been submitted:
53         if (isset($_POST['enterDetails'])) {
54
55             // Retrieve new username and password
56             // Save in session variables
57             $_SESSION["username"] = $_POST["username"];
58             $_SESSION["password"] = $_POST["password"];
59
60             // Login or register depending on loginType
61           if ($_SESSION["loginType"] == "Login") {
62                 // Login
63                 loginCheck();
64           }else{
65                 // Retrive password validation
66                 $valid = $_POST["valid"];
67                 // If password is valid, register new user
68                 if ($valid=="true") {
69                     registerCheck();
70                 }
71             }
72         }
73
74         // If play button has been pressed:
75         if (isset($_POST['scale'])) {
76             // Retrive start note and scale type
77             getScaleInfo();
78             // Play the scale with these values
79             echo "<script>playScale(".$_SESSION["startNote"].",".$_SESSION["scaleType"].");
   </script>";
80         }
81
82         // If the favourite button has been pressed
83         if (isset($_POST['favourite'])) {
84             // Retrive start note and scale type
85             getScaleInfo();
86             // Use these values to add new row to favourites table
87             addFavourite();
88         }
89
90         // If logout button has been pressed, set session variable to true
91         // Submit button refreshes page, the session variable is checked on load
92         if (isset($_POST['logout'])) {
93             $SESSION["logout"] = "true";
94         }
95     }
96 }
97
98 // Update entire page
99 // Takes place after checking all inputs and session variables
100 updatePage();
101
102 ?>
```

Log of On-going Testing

EXPECTED OUTCOME	**ERROR(S)**	**SOLUTION**	**REFERENCES**
The piano keys are the correct size	The black and white keys were all the same size and same colour.	Decided to create the piano key elements in a JavaScript loop. This way every key can be assigned an index, which can be used to check if the key should be white or black.	
The keyboard is laid out like a normal piano keyboard.	Found that all the keys stacked on top of each other on the left.	Implemented an indent style for all the keys. This is assigned to each key based on its index.	
Black keys are on top of the white keys	Black keys were not visible	Used CSS z-index to stack the elements in the correct order.	
When screen width is smaller than 720px, there should be two smaller keyboards stacked vertically.	No visible change	Used an event listener in JavaScript to trigger when the window is resized. When it crosses the 720px threshold in either direction, it destroys the entire keyboard and creates it correctly.	
The screen element S2 should look nice at all browser sizes.	The text of the scale name was overlapping the border.	Increased the width for the screen's media query to 1000px.	

It should be obvious if the user is using the login form or the register form.	It was unclear which one had been selected.	Added two small indicator lights next to the 'Login' and 'Register' buttons to clarify which button has been clicked.	
The appropriate forms should become inaccessible to a user depending on if they are logged in or not.	The inputs could still be used when they were meant to be inaccessible	Used JavaScript disabled to remove functionality of the correct elements.	https://www.w3schools.com/jsref/prop_select_disabled.asp
The user should not be able to log on or register whilst already registered	No errors		
When a scale is deleted from the favourites table, it should only remove it for the logged in user.	All instances of one scale were deleted from the favourites table.	Added a condition so that the username is also used to choose which scale to delete.	

Development of New Skills and Knowledge

NEW SKILL / KNOWLEDGE	PURPOSE	REFERENCES
To create a new element using JavaScript	Used to create the piano keys one by one in a loop. Also used to create audio elements when a key is played.	https://www.w3schools.com/jsref/met_document_createelement.asp
To add an 'onclick' listener in JavaScript and add it to an element.	Used to make all the piano keys trigger a JavaScript function when they are clicked.	https://www.w3schools.com/jsref/met_element_setattribute.asp
To delete an element from the website using JavaScript	When the keyboard needs to be created again because of a media query, used to delete all previous piano keys. Also used to delete an audio element after a key has been played. Deleting elements that have completed their purpose increases performance.	https://www.w3schools.com/jsref/met_element_remove.asp
To create an event listener that triggers a JavaScript function when the window is resized.	Used to make JavaScript media queries. This is used to re-create the keyboard when the width crosses a threshold.	https://www.w3schools.com/howto/howto_js_media_queries.asp
To be able to wait for a specific amount of time before triggering a JavaScript function.	Used to only play a key for 0.5s. Also used to time the intervals between notes when a scale is being played.	https://www.w3schools.com/jsref/met_win_settimeout.asp
To use 'localStorage' in JavaScript to keep a value even after the page refreshes.	Used to store a Boolean that indicates if the login type has changed. The JavaScript is cut off when the page refreshes, but the state of this value can still be checked after a refresh.	https://www.w3schools.com/jsref/prop_win_localstorage.asp
To use metacharacters to validate a string in JavaScript.	Used to validate the password when a user is registering to the site.	https://www.w3schools.com/jsref/jsref_obj_regexp.asp

Testing

To test my website, I will begin by creating a full test plan. I will describe the watchpoints used to see if the background processes are working as expected. I will create tests for each of my functional and end-user requirements. I will also create personas and test cases for usability testing.

I will then conduct all my tests using my test plan, recording evidence, and making sure that all my requirements have been met. I will test the usability of my website by using scenarios to make sure all motivations for using my site can be satisfied.

I did link this video earlier in the document, but it is worth mentioning that it covers many tests and shows off many requirements in a short space of time:

https://www.youtube.com/watch?v=1-Fqnq-HtUw

PREFIXES IN THIS SECTION	**DESCRIPTION**
F	Referencing a Functional Requirement
U	Referencing an End-User Requirement
S	Used to identify an element, defined at the design stage.
W	Referencing a watchpoint to be used during the testing process.
T	Referencing a test case for usability testing.

Final Test Plan

LIST OF WATCHPOINTS	
INDEX	PURPOSE
W1	Return a value to indicate the name of a played audio file.
W2	Return a value to indicate whether to login or to register a user.
W3	Return the username that was received by PHP.
W4	Return the password that was received by PHP.
W5	Return the password that is stored in the 'users' table.
W6	Return a value to indicate if the user will be logged in or not.
W7	Return a value to indicate if the new password is valid.
W8	Return a value to indicate if the user has been added to the 'users' table or not.
W9	Return a value to indicate when the logout button is clicked.

FUNCTIONAL REQUIREMENTS TEST PLAN		
INDEX	**TEST**	**EXPECTED OUTPUT**
F1	Click a key on the keyboard.	Key should appear to press. Correct audio file should be played out loud. Watchpoint W1 will return the name of this file. The name of the note should be displayed on the screen (S4).
F2	Fill in the start note (S8) and scale type (S9). Press the play button (S11).	The correct scale should be played up and back down the keyboard. The name of the scale should be displayed on the screen (S3). If there is no data, there should be no output.
F3	Start the website. Do not log in.	If an input is disabled, it should be greyed out. If an input is active, it will be a yellow colour. Inputs that should be disabled: • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7) Inputs that should be active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Login (S5) • Register (S6)
F4	As a logged-out user, click 'Login' (S5).	Screen (S2) should be in Layout B. A username (S3 i.), password (S3 ii.) and submit (S3 iii.) input should become available to the user. Watchpoint W2 should return value 'Login'

F5	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Click 'Login' (S5). Fill in the username (S3 i.). Fill in the password (S3 ii.). Submit the form (S3 iii.).	Watchpoint W3 should return the value of the username entered. Watchpoint W4 should return the value of the password entered. Watchpoint W5 should return the value of the user's real password. Watchpoint W6 will return 'true' if the details are valid, or 'false' if an incorrect username or password is entered.
F6	As a logged-out user, click 'Register' (S6).	Screen (S2) should be in Layout C. A username (S3 i.), password (S3 ii.) and submit (S3 iii.) input should become available to the user. Password assistance should become available to the user in element S4. Watchpoint W2 should return value 'Register'
F7	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Click 'Register' (S6). Fill in the username (S3 i.). Fill in the password (S3 ii.). Submit the form (S3 iii.).	Watchpoint W3 should return the value of the username entered. Watchpoint W4 should return the value of the password entered. Watchpoint W7 will return 'true' if the password is valid, or 'false' if the password is invalid. Watchpoint W8 will return 'true' if the user has been registered, or 'false' if the user was not registered.

F8	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Start the website. Log in using the details above.	If an input is disabled, it should be greyed out. If an input is active, it will be a yellow colour. Inputs that should be disabled: • Login (S5) • Register (S6) Inputs that should be active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7)
F9	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Start the website. Log in using the details above. Fill in the start note (S8) and scale type (S9). Press the 'favourite' button (S10). Press the 'favourite' button (S10) again to unfavourite a scale.	If the scale is not favourited, it should go into the favourites list (S13). The scale should be added to the database. If the scale is already favourited, it should be removed from the favourites list (S13). The scale should be removed to the database. If a user logs out and back in, the list should remain in its previous state.

F10	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Start the website. Log in using the details above. This user will already have favourited these scales: • 'F Mixolydian' • 'D#/Eb Lydian' • 'A Major Pentatonic' Select a scale from the dropdown list (S13)	The favourite scale that was selected should be filled into the scale form.
F11	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Start the website. Log in using the details above. This user will already have favourited these scales: • 'F Mixolydian' • 'D#/Eb Lydian' • 'A Major Pentatonic' Fill in the start note (S8) and scale type (S9).	If the start note and scale type match a favourited scale, the heart icon in S10 should be filled. If the start note and scale type do not match a favourited scale, the heart icon in S10 should be hollow.

F12	The 'users' database will already contain a user with: • Username: 'callum' • Password: 'callum1!' Start the website. Log in using the details above. Click 'Logout' (S7).	If an input is disabled, it should be greyed out. If an input is active, it will be a yellow colour. Inputs that should be disabled: • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7) Inputs that should be active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Login (S5) • Register (S6) Element S2 should be in Layout A Watchpoint W9 should return value 'true'

END-USER REQUIREMENTS TEST PLAN		
INDEX	TEST CASE	EXPECTED OUTCOME
U1	Attempt to play a key on the keyboard. Click a key on the keyboard.	The user should be able to tell which key they have played. The user should be able to hear the note out loud.
U2	Attempt to choose a scale to play. Fill in the scale form with a start note and scale type.	The user will be able to fill in any start note or scale type to their liking.
U3	Attempt to listen to a scale. After filling in the scale form with a start note and scale type, press the play button.	The user will see and hear the correct scale being played back to them, up and down the scale.
U4	Attempt to find out the name of a note or the name of a scale. Play a key on the keyboard. Fill in the scale form with a start note and scale type.	The note played should show up as text on the right side of the screen. The selected scale should show up as text on the left side of the screen.
U5	Attempt to find a login or register form. Click 'Login' and 'Register'.	The 'Login' button should bring up the login form. The 'Register' button should bring up the register form and password validation.
U6	Attempt to create a valid password. Click 'Register' and begin to fill in the password field.	The password validation on the right hand of the screen should update as the user is typing the password.
U7	Attempt to register. Click the 'Register' button. Choose a username and a valid password. Fill in and submit the register form.	The user should be registered and logged into the website.
U8	Attempt to login. The user will already have a login. Click 'Login' and fill in the login form. Submit the form.	The user should be logged into the website.
U9	Attempt to favourite a scale. Log into the site. Fill in the scale form and click the heart icon.	The scale should now be in the favourites list to the right of the play button.
U10	Attempt to remove a scale from the favourites list. Log into the site. Select a favourited scale. Click the heart icon.	The scale should be removed from the favourites list on the right of the play button.

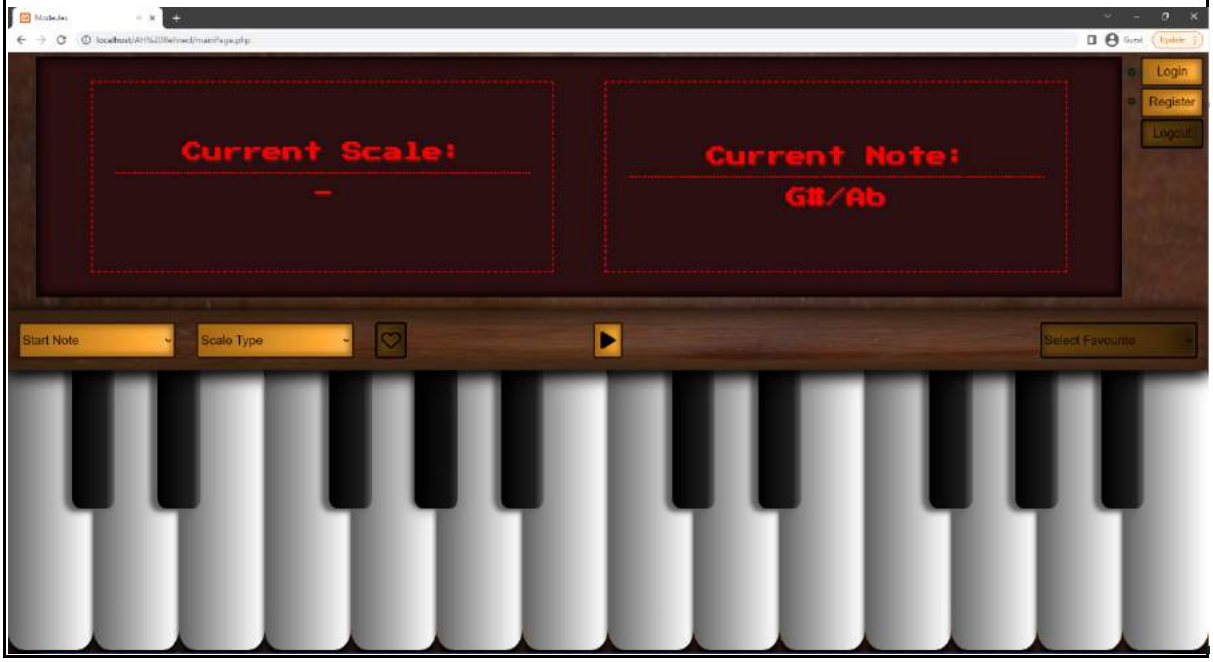
U11	Attempt to check if a scale has been favourited. Log into the site. Fill in the scale form and observe the heart icon.	The heart icon should be filled if the selected scale is a favourite. The heart icon should be hollow if the selected scale is not a favourite.
U12	Attempt to use a scale that has been favourited. Log into the site. Select the favourite list and choose an option.	The selected favourite should be filled into the scale form.
U13	Attempt to log out of the website. Log into the site. Click the 'Logout' button.	The user should be logged out of the website and the site should revert to its default state.

PERSONAS	
PERSONA	**DESCRIPTION**
Inexperienced User	A user who has little experience of browsing a website. This user my struggle to spot inputs.
Experienced User	A user who has a lot of experience of browsing a website. They should have no trouble filling in a form.

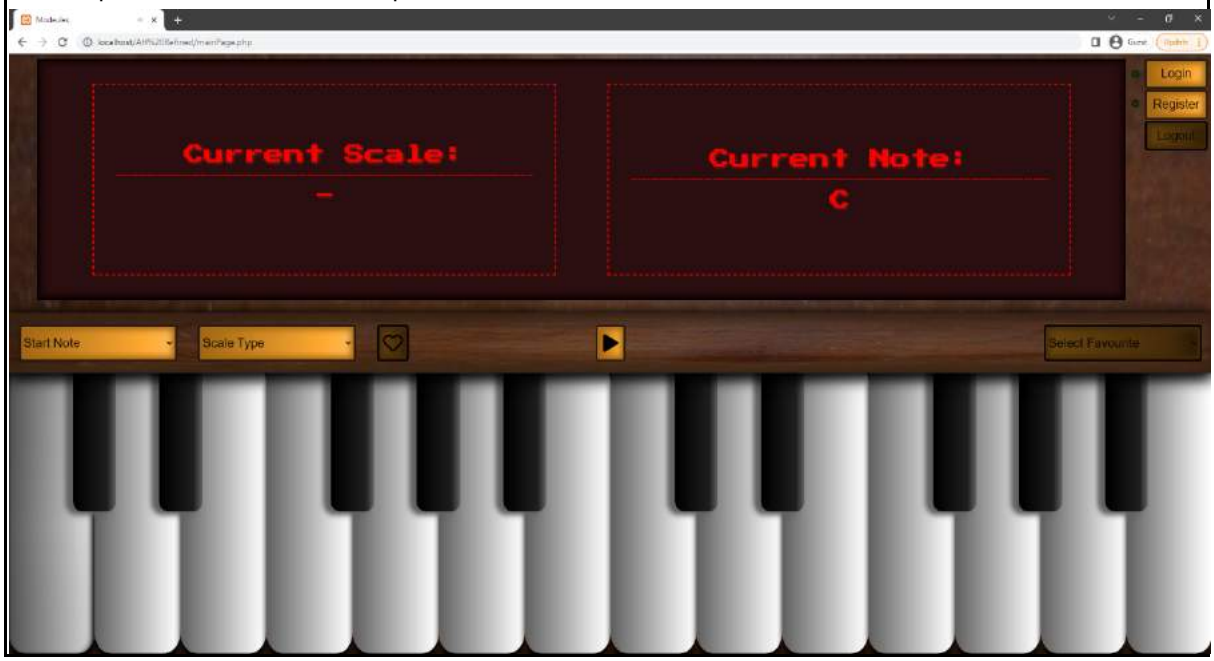
TEST CASES		
INDEX	**TEST CASE**	**DESCRIPTION**
T1	Users playing the keyboard	The user will be asked to play the keyboard.
T2	Users playing a scale	The user will be asked to pick a scale to play, and then to play it.
T3	Users registering	The user will be asked to create an account, and to log in.
T4	Users logging on	The user will be asked to log on to an account.
T5	Users favouriting a scale	The user will be asked to pick a scale and to add it to their favourites.
T6	Users logging off	The user will be asked to log out of an account.

Functional Requirements Testing

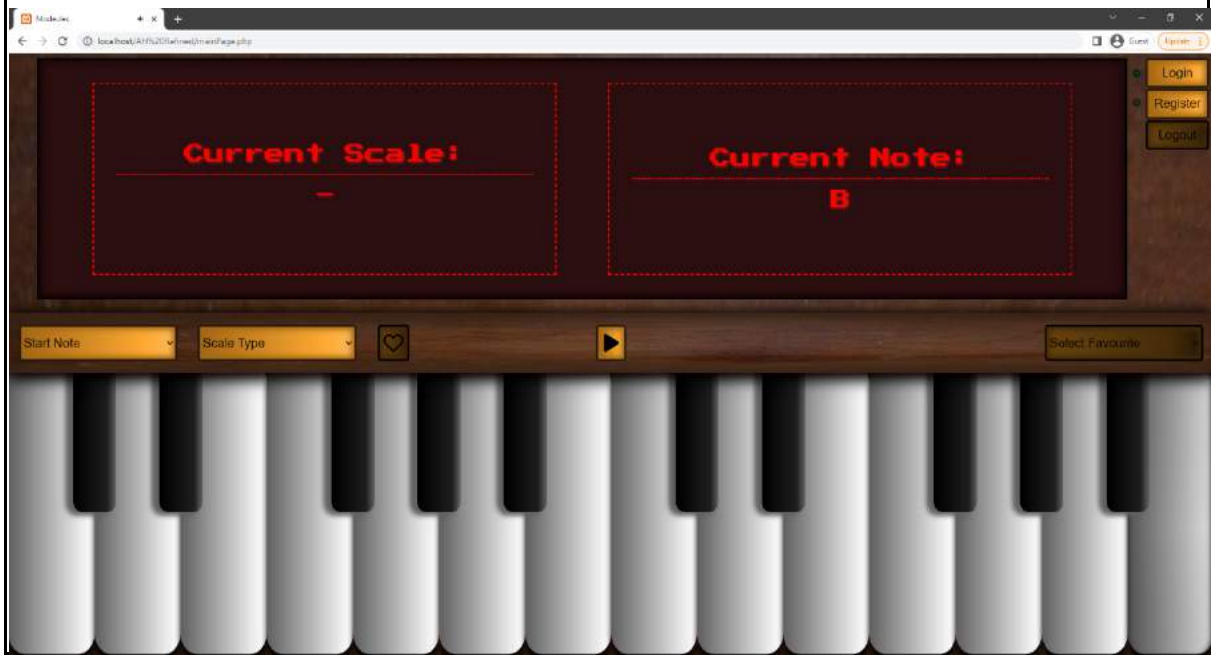
F1		
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Normal Test: Clicked G#/Ab	Audible G#/Ab note. Key pressing animation. "G#/Ab" in text on the screen	✓
EVIDENCE		
Watchpoint W1 Returned 'Ab1.mp3'		



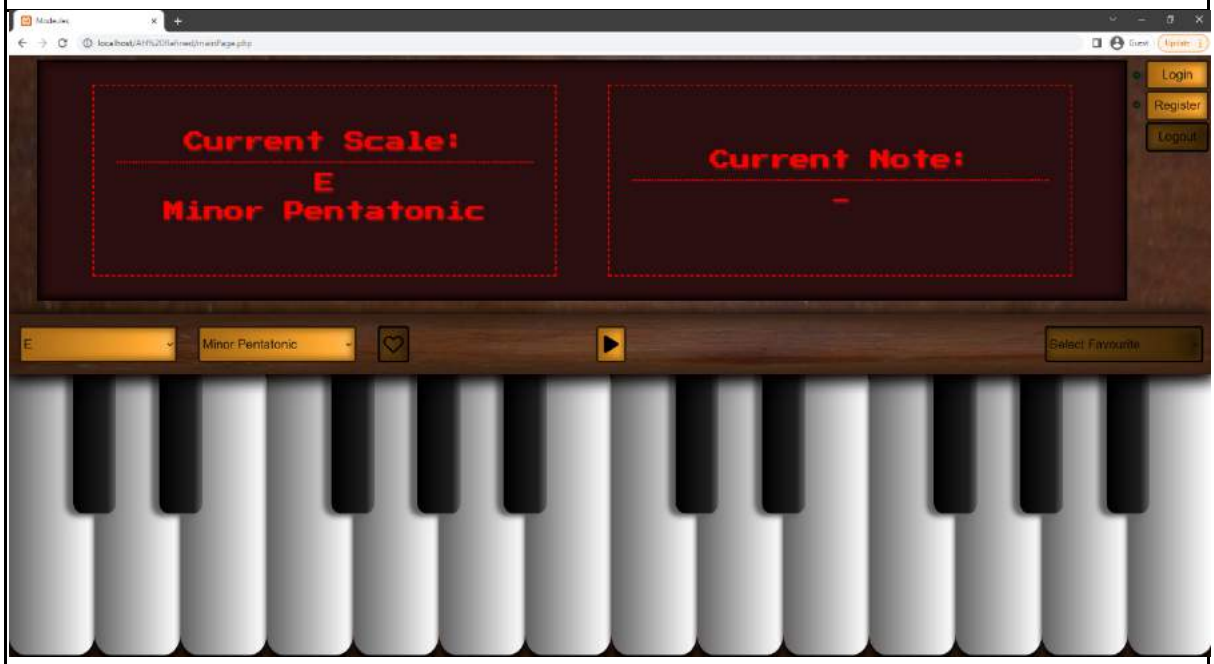
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Extreme Test: Clicked first key (C1)	Audible C note. Key pressing animation. "C" in text on the screen	✓
EVIDENCE		
Watchpoint W1 Returned 'C1.mp3'		



TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Extreme Test: Clicked last key (B2)	Audible B note. Key pressing animation. "B" in text on the screen	✓
EVIDENCE		
Watchpoint W1 Returned 'B2.mp3'		

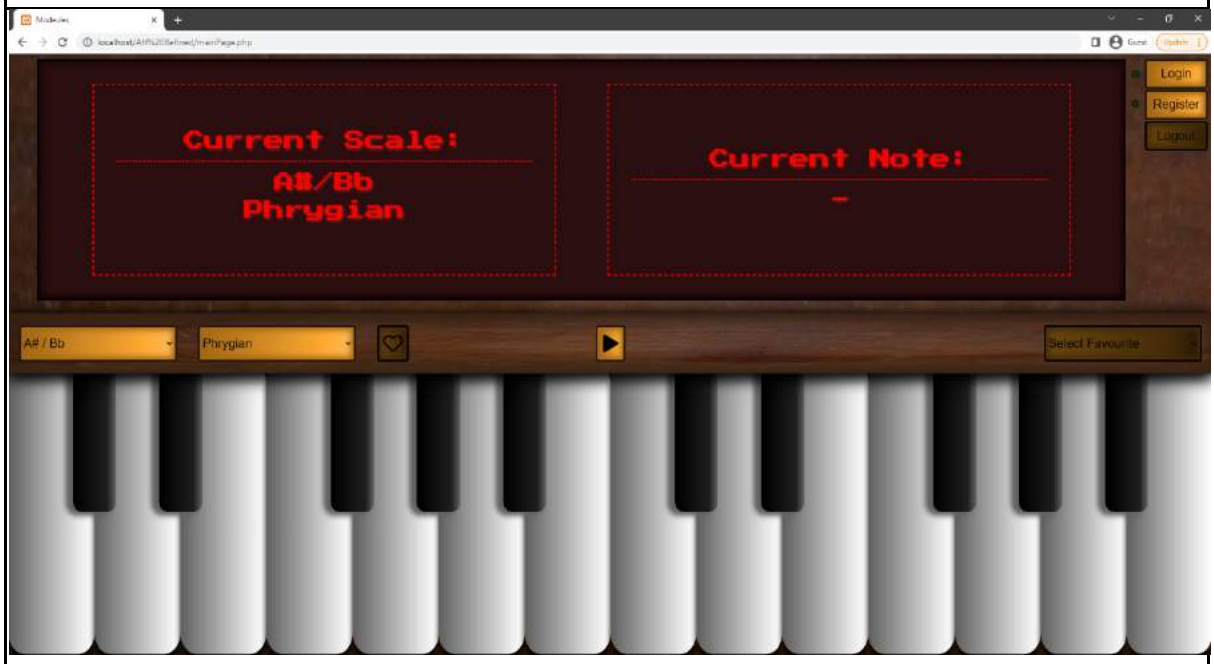


F2		
TEST DATA	ACTUAL OUTPUT	RESULT
Normal Test: Start Note: E Scale Type: Minor Pentatonic	Keys played in order and then in reverse: E,G,A,B,D,E Text should be displayed: "E Minor Pentatonic"	✓
EVIDENCE		

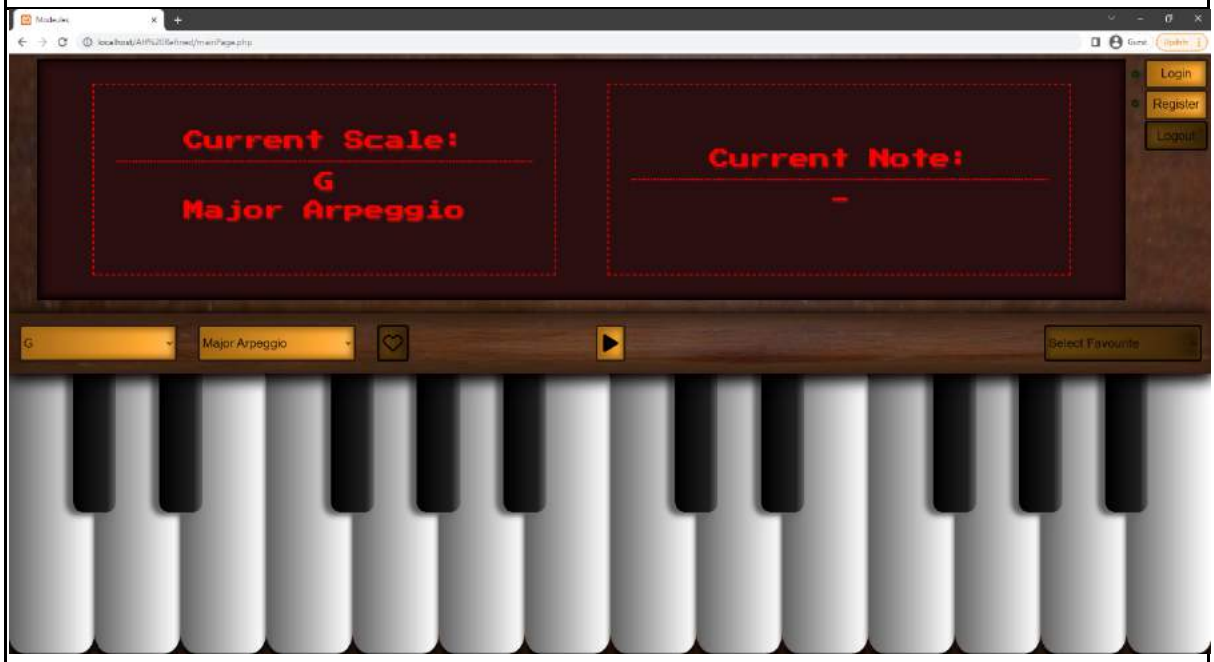


TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Normal Test: Start Note: A#/Bb Scale Type: Phrygian	Keys played in order and then in reverse: Bb,B,Db,Eb,F,Gb,Ab,Bb Text should be displayed: "A#/Bb Phryian"	✓

EVIDENCE

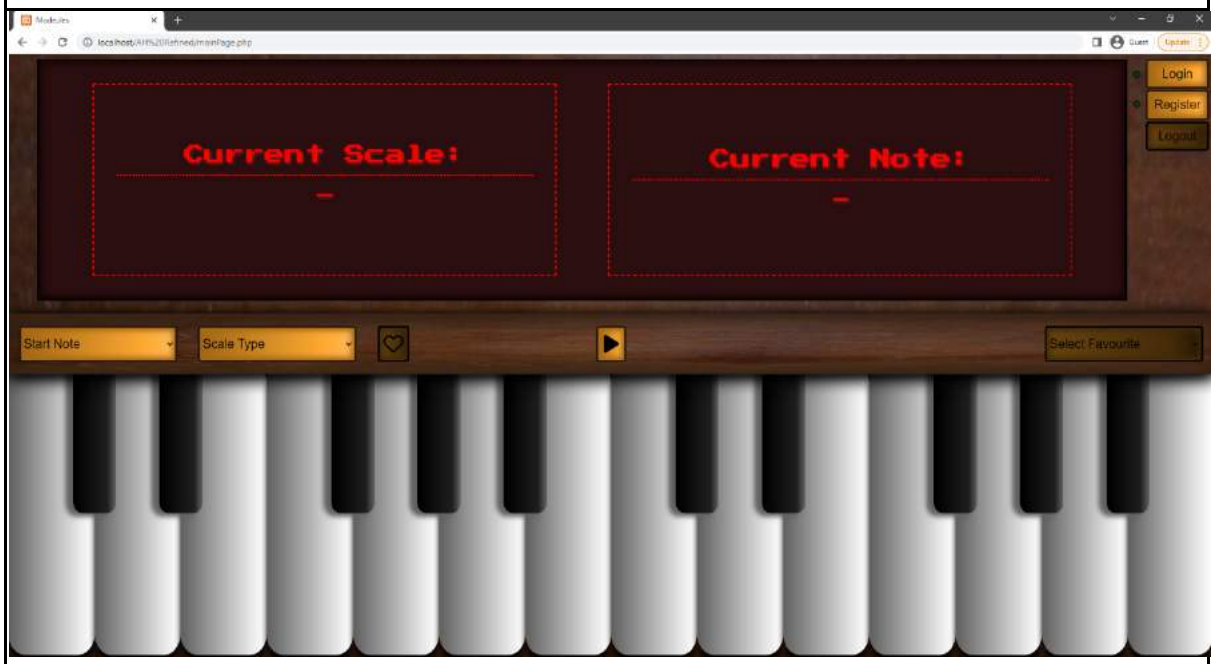


TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Normal Test: Start Note: G Scale Type: Major Arpeggio	Keys played in order and then in reverse: G,B,D,G Text should be displayed: "G Major Arpeggio"	✓
EVIDENCE		



TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Test: Start Note: Default Scale Type: Default	No output	✓

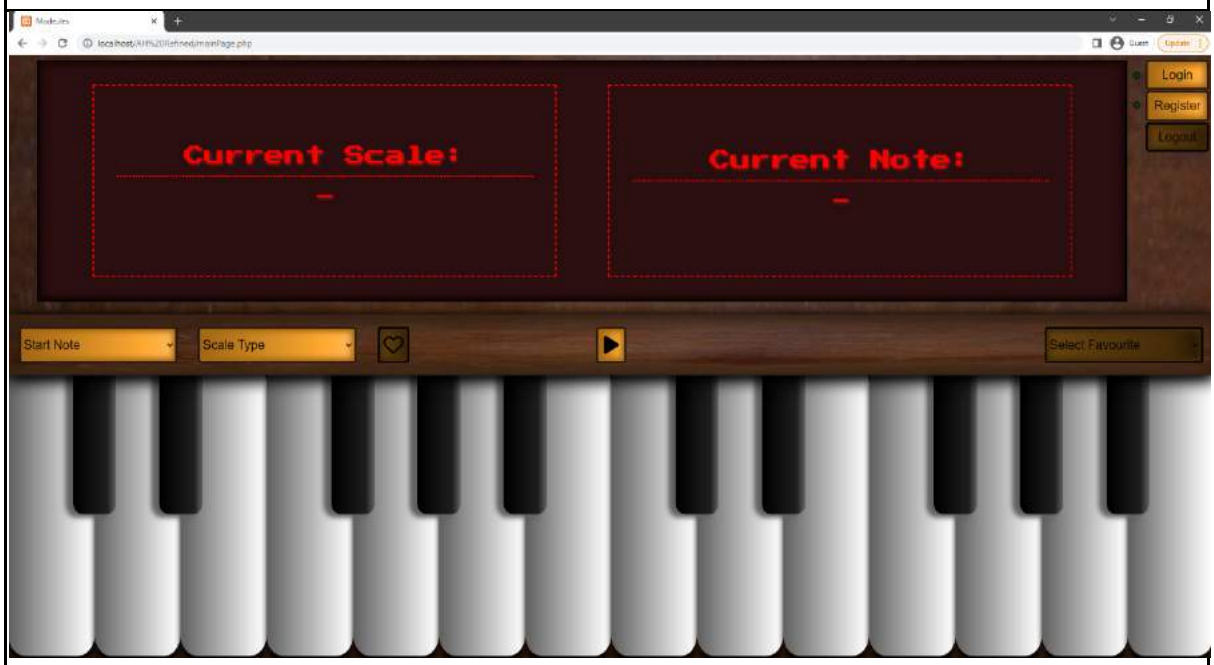
EVIDENCE



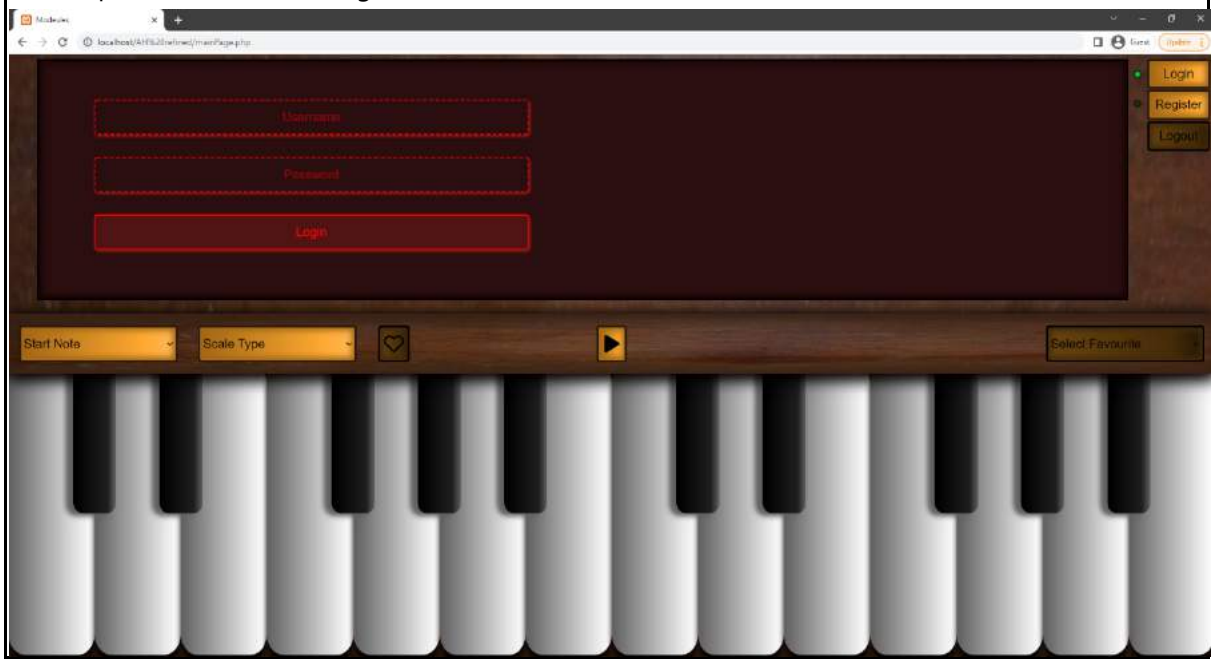
F3

TEST DATA	ACTUAL OUTPUT	RESULT
Start the website up.	Inputs that were disabled: • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7) Inputs that were active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Login (S5) • Register (S6)	✓

EVIDENCE



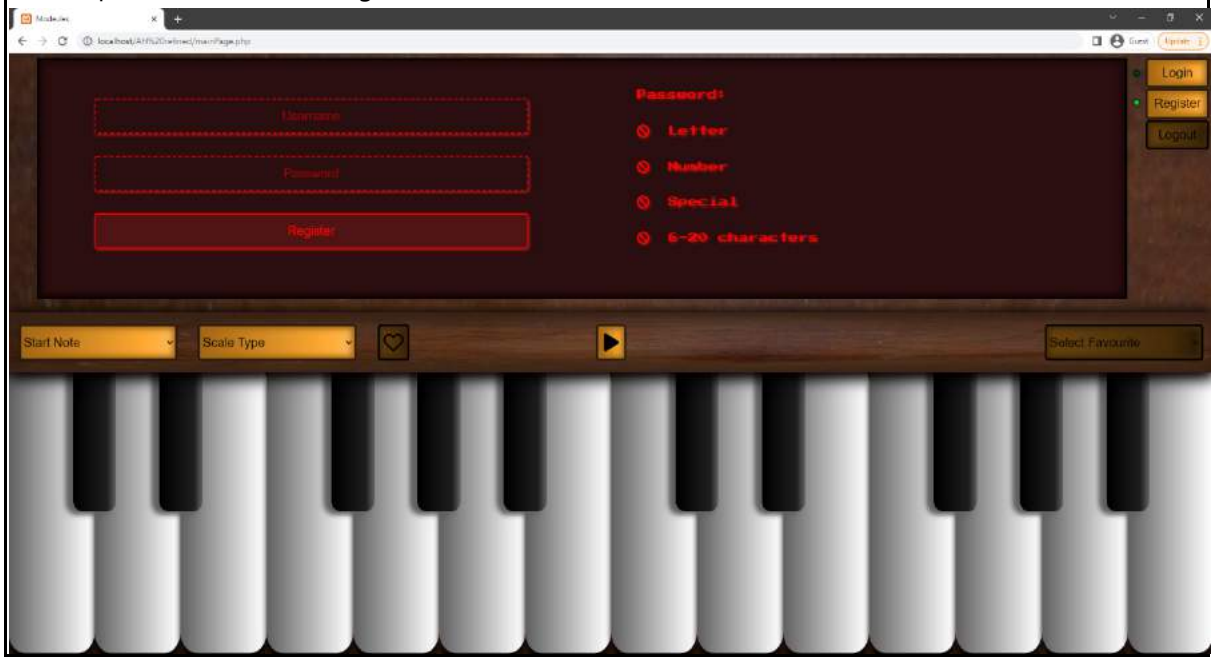
F4		
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Start the website up. Click 'Login' (S5)	Screen (S2) changes to Layout B. A username (S3 i.), password (S3 ii.) and submit (S3 iii.) input appear in the screen element (S3).	✓
EVIDENCE		
Watchpoint W2 returned 'Login'		



F5		
TEST DATA	ACTUAL OUTPUT	RESULT
Normal Data Username: callum Password: callum1!	Username and password were collected from the form. The details were compared to the database and the user was logged in.	✓
EVIDENCE		
Watchpoint W3 returned 'callum' Watchpoint W4 returned 'callum1!' Watchpoint W5 returned 'callum1!' Watchpoint W6 returned 'true'		
TEST DATA	ACTUAL OUTPUT	RESULT
Exceptional Data Username: invalid Password: callum1!	Username and password were collected from the form. The details were compared to the database and the user was not logged in due to an invalid username.	✓
EVIDENCE		
Watchpoint W3 returned 'invalid' Watchpoint W4 returned 'callum1!' Watchpoint W5 returned Null Watchpoint W6 returned 'false'		

TEST DATA	ACTUAL OUTPUT	RESULT
Exceptional Data Username: callum Password: invalid	Username and password were collected from the form. The details were compared to the database and the user was not logged in due to an invalid password.	✓
EVIDENCE		
Watchpoint W3 returned 'callum' Watchpoint W4 returned 'invalid' Watchpoint W5 returned 'callum1!' Watchpoint W6 returned 'false'		
TEST DATA	ACTUAL OUTPUT	RESULT
Exceptional Data Username: invalid1 Password: invalid2	Username and password were collected from the form. The details were compared to the database and the user was not logged in due to an invalid username and password.	✓
EVIDENCE		
Watchpoint W3 returned 'invalid1' Watchpoint W4 returned 'invalid2' Watchpoint W5 returned Null Watchpoint W6 returned 'false'		

F6		
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Start the website up. Click 'Register' (S6)	Screen (S2) changes to Layout C. A username (S3 i.), password (S3 ii.) and submit (S3 iii.) input appear in the screen element (S3). Password assistance appears in element S4.	✓
EVIDENCE		
Watchpoint W2 returned 'Register'		



F7

TEST DATA	ACTUAL OUTPUT	RESULT
Normal Data Username: newUN Password: newPW1!	Username and password were collected from the form. The password was valid, so the new user was added to the database.	✓

EVIDENCE

Watchpoint W3 returned 'newUN'

Watchpoint W4 returned 'newPW1!'

Watchpoint W7 returned 'true'

Watchpoint W8 returned 'true'

Database:

username	password
callum	callum1!
newUN	newPW1!

TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Data Username: callum Password: callum1!	Username and password were collected from the form. The password was valid, but the user already exists in the database. The new user was not added to the database.	✓

EVIDENCE

Watchpoint W3 returned 'callum'
Watchpoint W4 returned 'callum1!'
Watchpoint W7 returned 'true'
Watchpoint W8 returned 'false'
No change to database.

TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Data Username: newUN Password: newPW1	Username and password were collected from the form. The password was invalid, so the new user was not added to the database.	✓

EVIDENCE

Watchpoint W3 returned 'newUN'
Watchpoint W4 returned 'newPW1'
Watchpoint W7 returned 'false'
Watchpoint W8 returned 'false'
No change to database.

TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Data Username: newUN Password: newPW!	Username and password were collected from the form. The password was invalid, so the new user was not added to the database.	✓
EVIDENCE		
Watchpoint W3 returned 'newUN' Watchpoint W4 returned 'newPW!' Watchpoint W7 returned 'false' Watchpoint W8 returned 'false' No change to database.		

TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Data Username: newUN Password: 111!!!	Username and password were collected from the form. The password was invalid, so the new user was not added to the database.	✓

EVIDENCE

Watchpoint W3 returned 'newUN'
Watchpoint W4 returned '111!!!'
Watchpoint W7 returned 'false'
Watchpoint W8 returned 'false'
No change to database.

TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Exceptional Data Username: newUN Password: short	Username and password were collected from the form. The password was invalid, so the new user was not added to the database.	✓

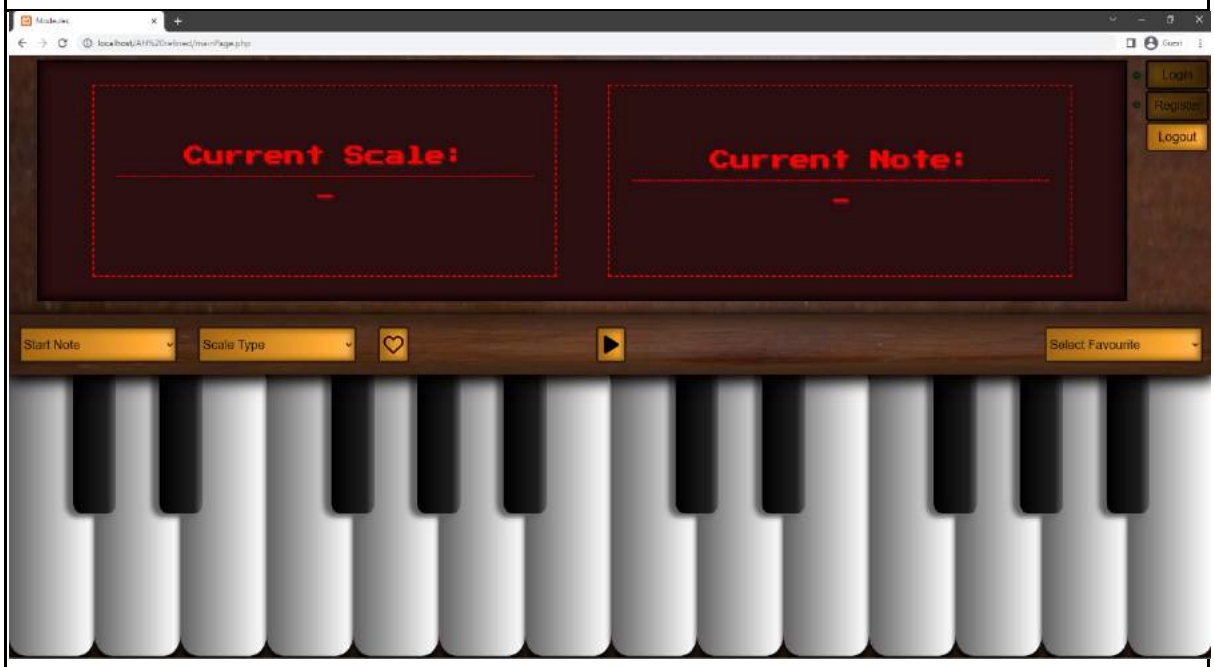
EVIDENCE

Watchpoint W3 returned 'newUN'
Watchpoint W4 returned 'short'
Watchpoint W7 returned 'false'
Watchpoint W8 returned 'false'
No change to database.

F8

TEST DATA	ACTUAL OUTPUT	RESULT
Start the website up. Login with details: • Username: callum • Password: callum1!	Inputs that were disabled: • Login (S5) • Register (S6) Inputs that were active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7)	✓

EVIDENCE



F9

TEST DATA	ACTUAL OUTPUT	RESULT
Favourite 'F Mixolydian'	'F Mixolydian' was added to favourites.	✓

EVIDENCE

Database:

username	startNote	scaleType
callum	5	4

TEST DATA	ACTUAL OUTPUT	RESULT
Favourite 'D#/Eb Lydian'	'D#/Eb Lydian' was added to favourites.	✓

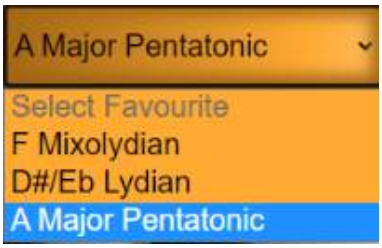
EVIDENCE

Database:

username	startNote	scaleType
callum	5	4
callum	3	3

TEST DATA	ACTUAL OUTPUT	RESULT
Favourite 'A Major Pentatonic'	'A Major Pentatonic' was added to favourites.	✓

EVIDENCE



Database:

username	startNote	scaleType
callum	5	4
callum	3	3
callum	9	7

TEST DATA	ACTUAL OUTPUT	RESULT
Unfavourite 'D#/Eb Lydian'	'D#/Eb Lydian' was removed from favourites.	✓

EVIDENCE

Database:

username	startNote	scaleType
callum	5	4
callum	9	7

TEST DATA	ACTUAL OUTPUT	RESULT
Log out and back in.	The favourites list remained the same after logging out and back in.	✓

EVIDENCE

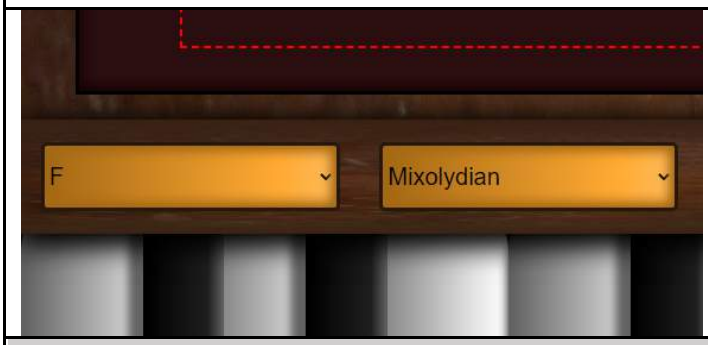
Database:

username	startNote	scaleType
callum	5	4
callum	9	7

F10

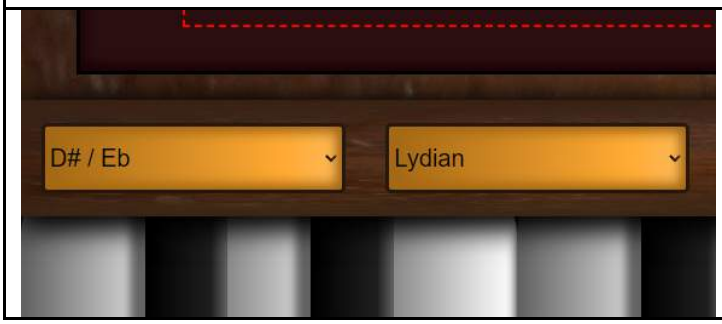
TEST DATA	ACTUAL OUTPUT	RESULT
Select 'F Mixolydian'	'F Mixolydian' was filled into the scale form.	✓

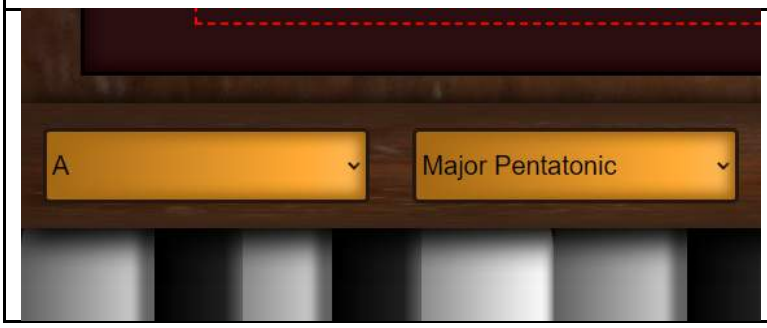
EVIDENCE



TEST DATA	ACTUAL OUTPUT	RESULT
Select 'D#/Eb Lydian'	'D#/Eb Lydian' was filled into the scale form.	✓

EVIDENCE

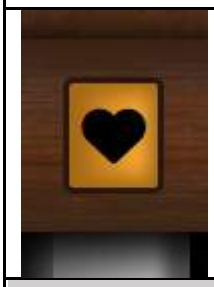


TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Select 'A Major Pentatonic'	'A Major Pentatonic' was filled into the scale form.	✓
EVIDENCE		
		

F11

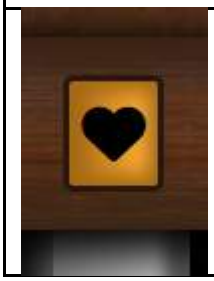
TEST DATA	ACTUAL OUTPUT	RESULT
Start note: F Scale type: Mixolydian	Heart icon was filled.	✓

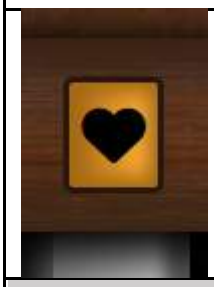
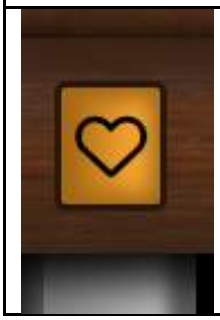
EVIDENCE



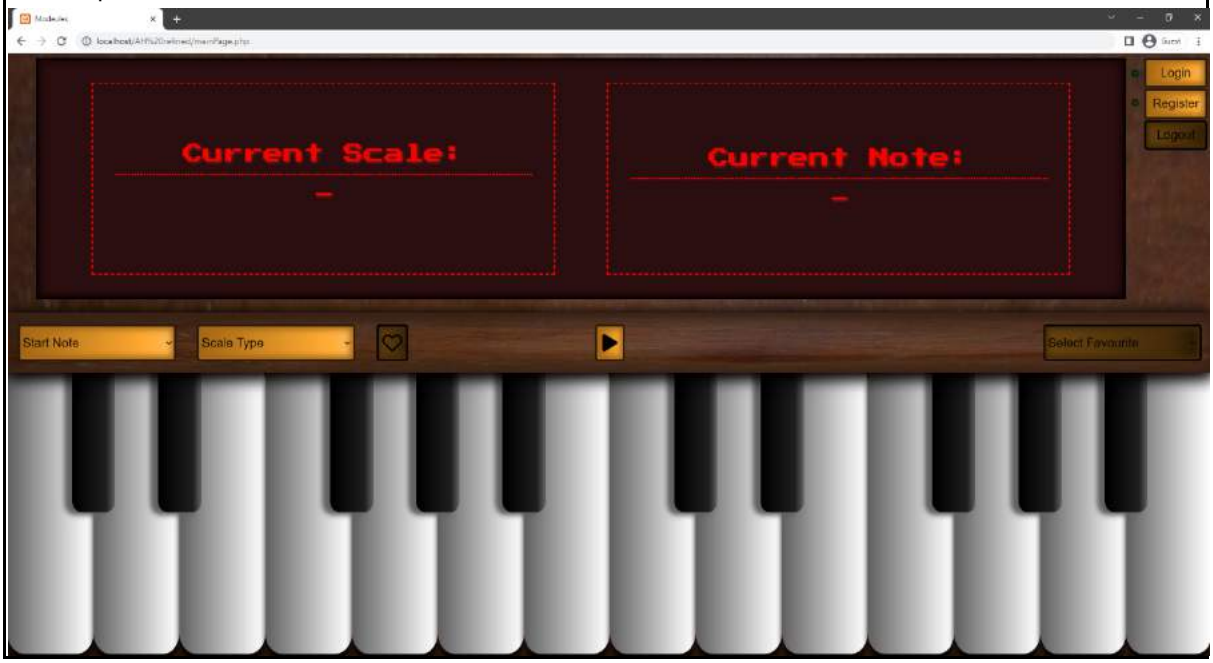
TEST DATA	ACTUAL OUTPUT	RESULT
Start note: D#/Eb Scale type: Lydian	Heart icon was filled.	✓

EVIDENCE



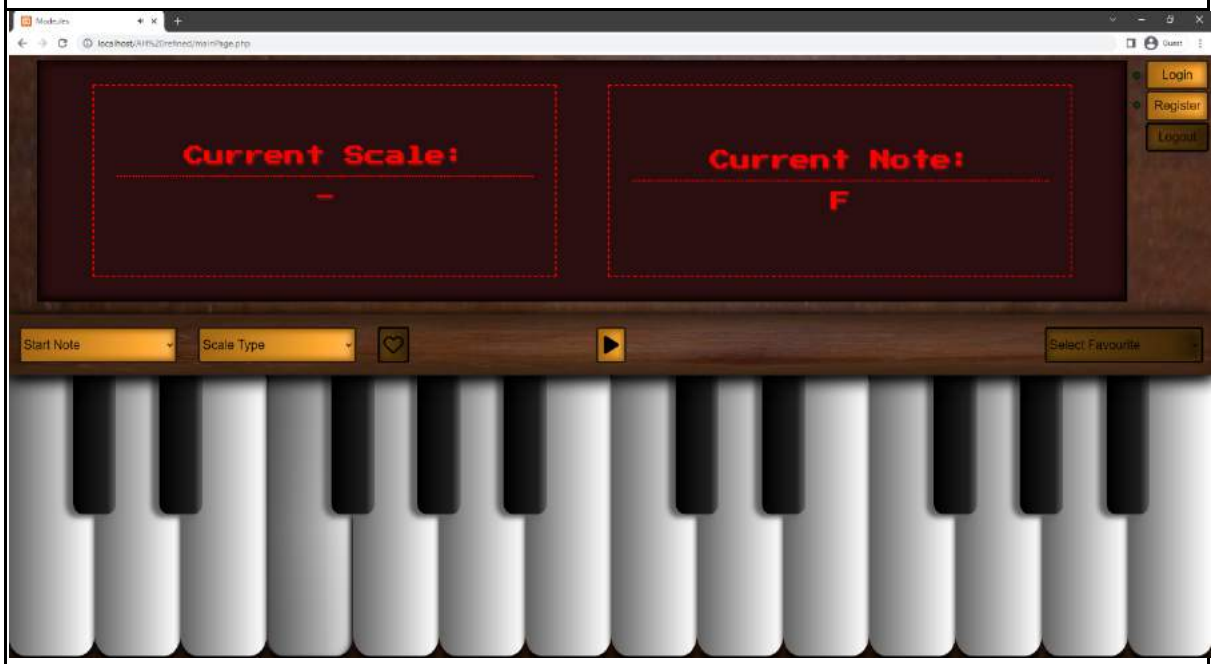
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Start note: A Scale type: Major Pentatonic	Heart icon was filled.	✓
EVIDENCE		
		
TEST DATA	**ACTUAL OUTPUT**	**RESULT**
Start note: C Scale type: Lydian	Heart icon was hollow.	✓
EVIDENCE		
		

F11		
TEST DATA	ACTUAL OUTPUT	RESULT
Start the website up. Login with details: • Username: callum • Password: callum1! Click 'Logout' (S7)	Inputs that were disabled: • Favourite Button (S10) • Choose Favourite Input (S13) • Logout Button (S7) Inputs that were active: • Start Note (S8) • Scale Type (S9) • Play Button (S11) • Login (S5) • Register (S6) Element S2 was changed to Layout A	✓
EVIDENCE		
Watchpoint W9 returned 'true'		



User Requirements Testing

U1		
TEST INPUT	ACTUAL OUTPUT	RESULT
Click ‘F’ key.	‘F’ key was pressed, heard and displayed as text on the screen.	✓
EVIDENCE		

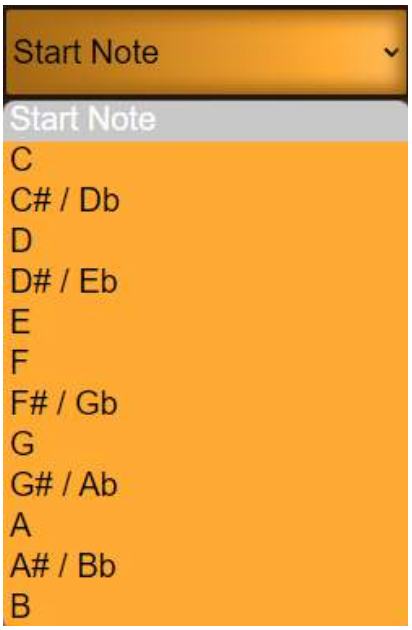


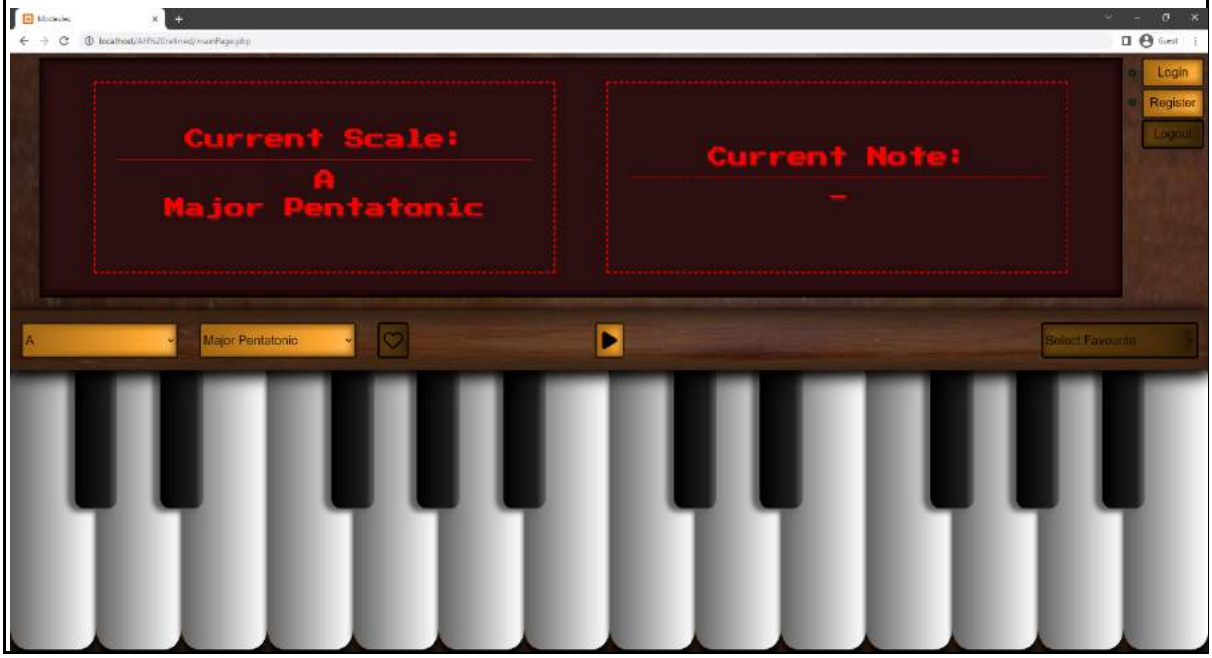
U2

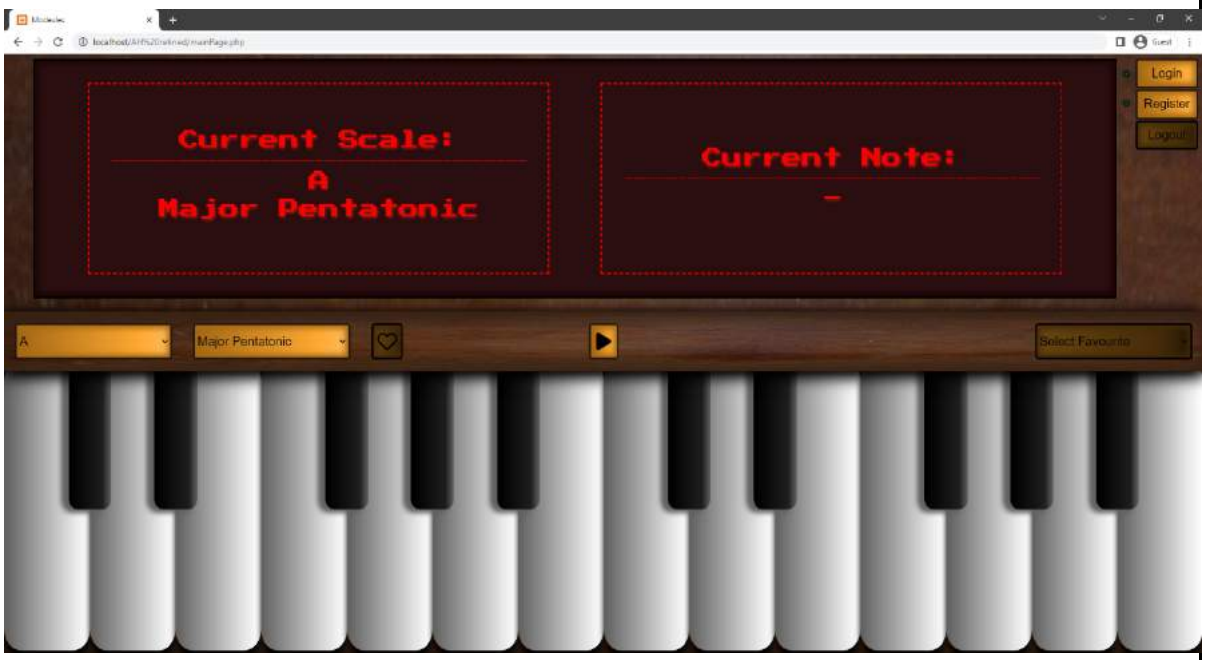
TEST INPUT	ACTUAL OUTPUT	RESULT
Fill the scale form in with: Start Note: A Scale Type: Major Pentatonic	The options were available to the user.	✓

EVIDENCE

Options available:



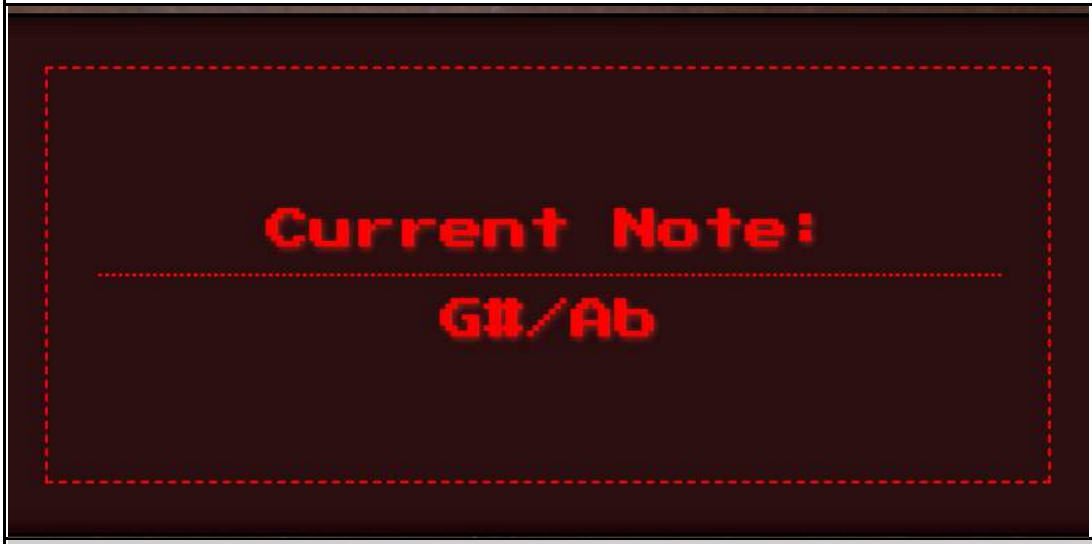


U3		
TEST INPUT	ACTUAL OUTPUT	RESULT
Fill the scale form in with: Start Note: A Scale Type: Major Pentatonic Click the 'Play' button.	The scale was played up and down the keyboard. The user was able to hear and see every note being played.	✓
EVIDENCE		
		

U4

TEST INPUT	ACTUAL OUTPUT	RESULT
Click the G#/Ab key.	The name of the note (G#/Ab) was displayed as text.	✓

EVIDENCE



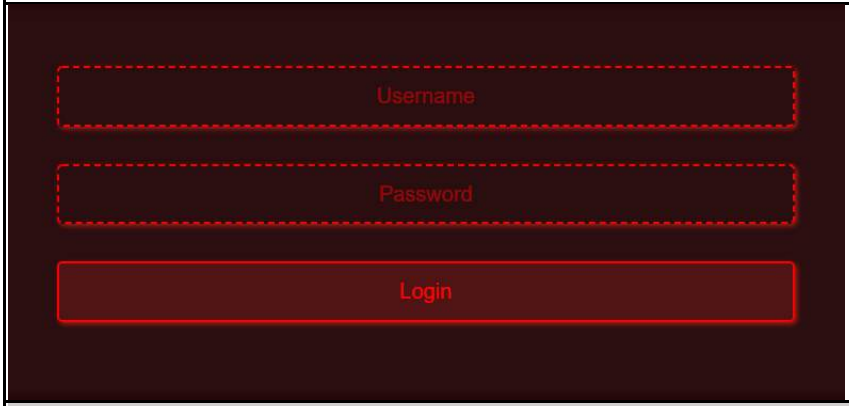
TEST INPUT	ACTUAL OUTPUT	RESULT
Fill the scale form in with: Start Note: E Scale Type: Locrian	The name of the scale (E Locrian) was displayed as text.	✓

EVIDENCE

U5

TEST INPUT	ACTUAL OUTPUT	RESULT
Click the 'Login' button.	The login form appeared.	✓

EVIDENCE

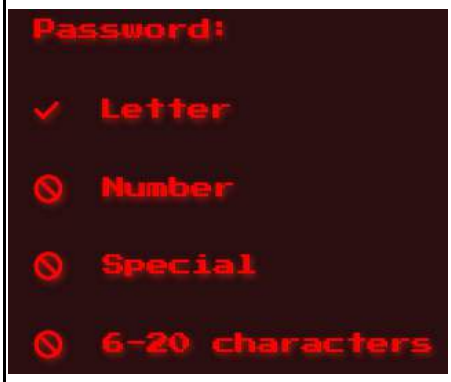


TEST INPUT	ACTUAL OUTPUT	RESULT
Click the 'Register' button.	The register form and password validation appeared.	✓

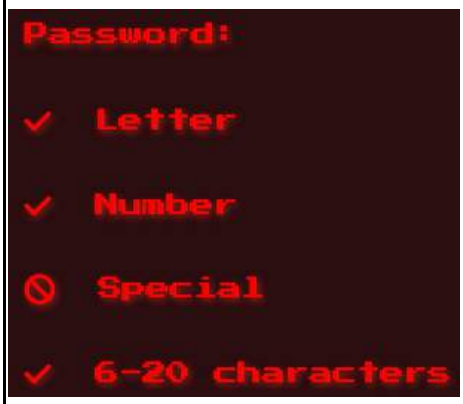
EVIDENCE

U6		
TEST INPUT	ACTUAL OUTPUT	RESULT
Fill in the password with various degrees of validity.	The password validation was always up to date, giving feedback to the user.	✓
EVIDENCE		

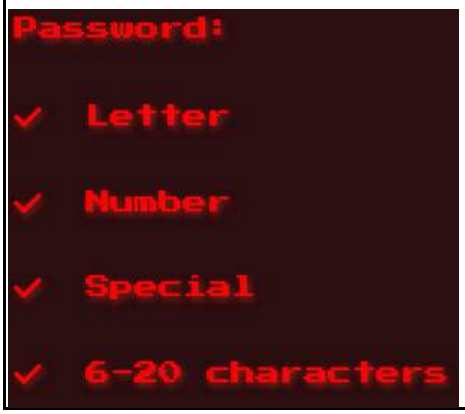
Password: 'hello'



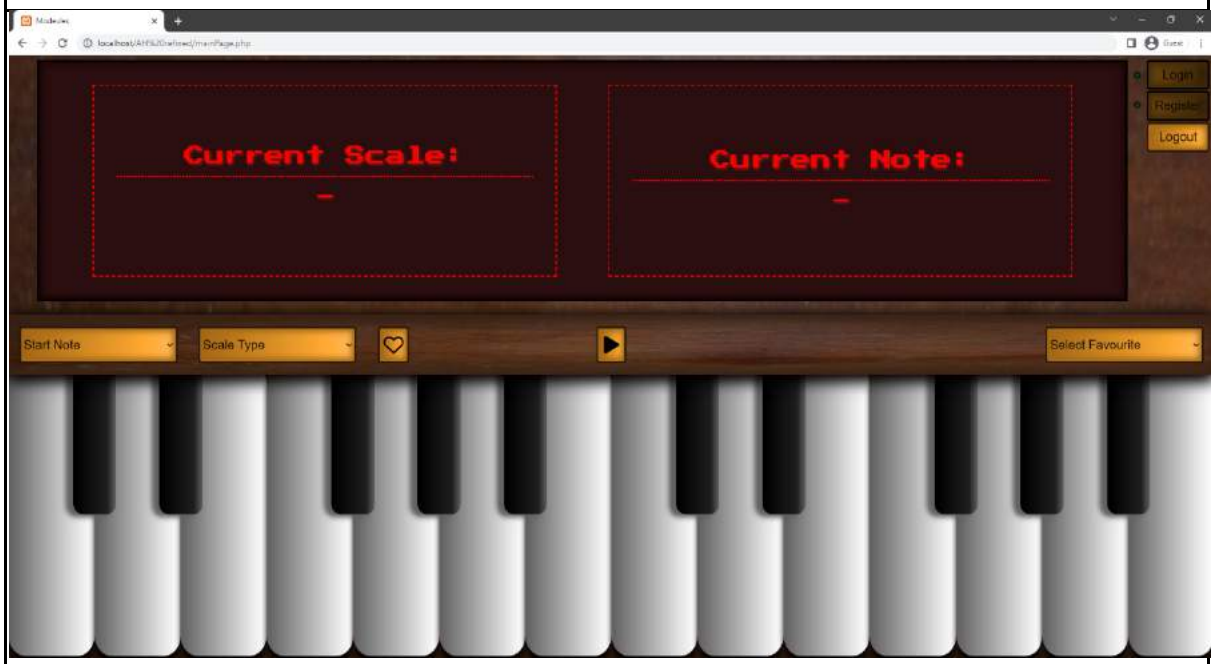
Password: 'hello1'



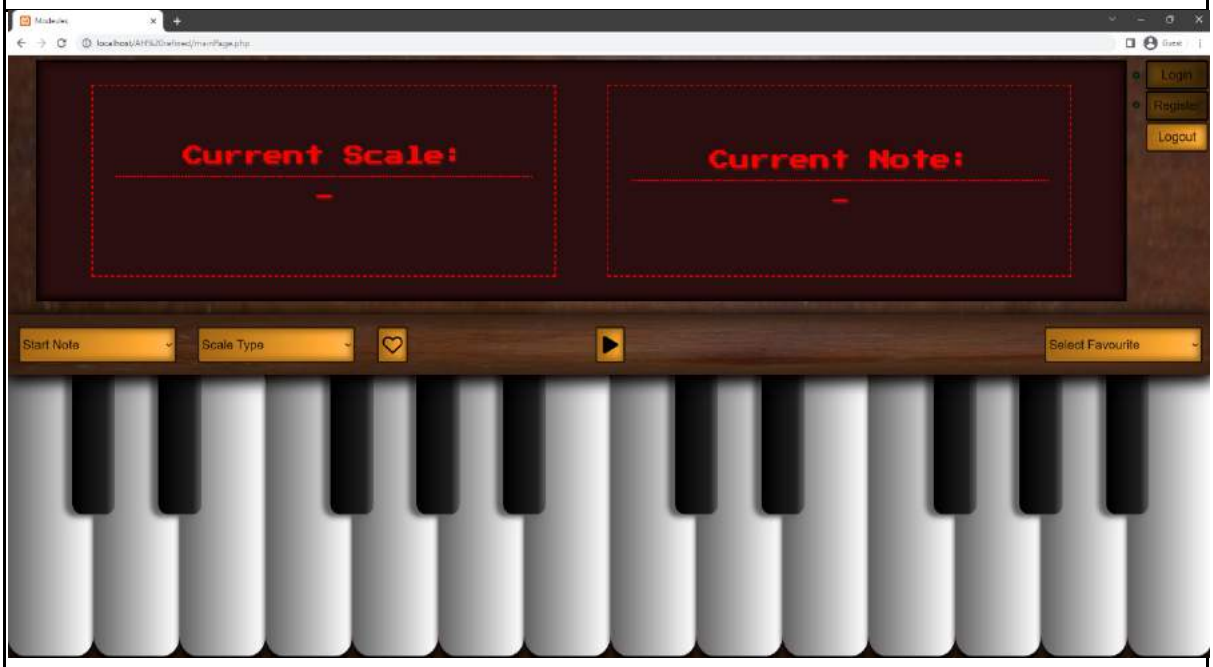
Password: 'hello1?'



U7		
TEST INPUT	**ACTUAL OUTPUT**	**RESULT**
Fill in register form with: Username: newLogin Password: hello1? Submit the form.	The user was logged in and gained access to the favourite inputs.	✓
EVIDENCE		

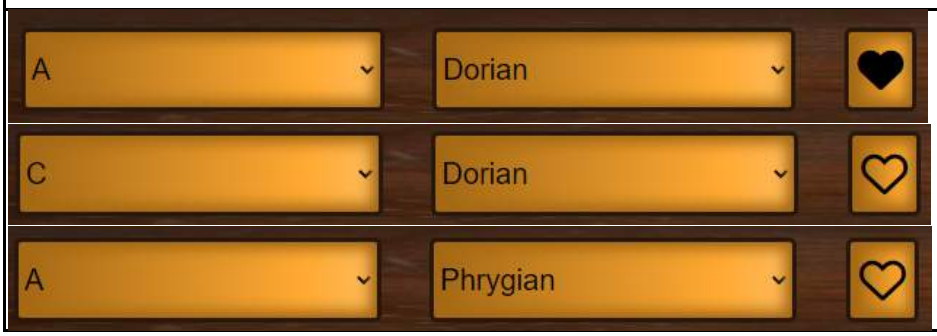


U8		
TEST INPUT	ACTUAL OUTPUT	RESULT
Fill in login form with: Username: callum Password: callum1! Submit the form.	The user was logged in and gained access to the favourite inputs.	✓
EVIDENCE		



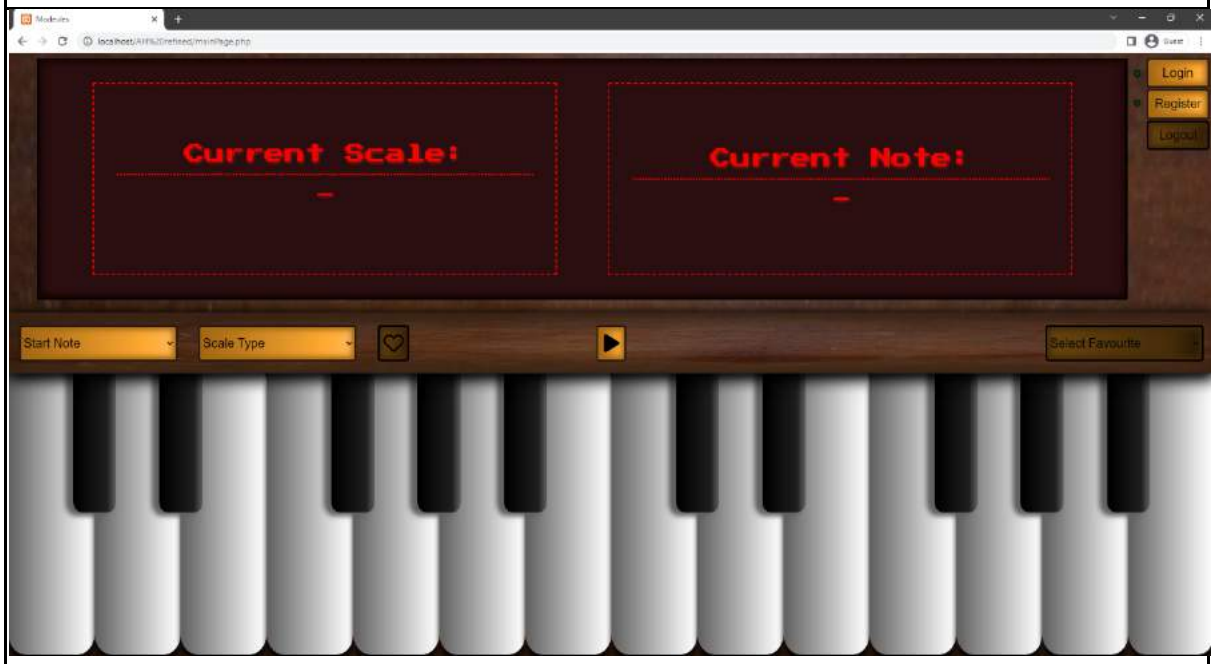
U9		
TEST INPUT	ACTUAL OUTPUT	RESULT
Favourite 'D Minor Arpeggio'	The scale 'D Minor Arpeggio' was added to the favourites of this user.	✓
EVIDENCE		

U10		
TEST INPUT	ACTUAL OUTPUT	RESULT
Unfavourite 'D Minor Arpeggio'	The scale 'D Minor Arpeggio' was removed from the favourites of this user.	✓
EVIDENCE		

U10		
TEST INPUT	**ACTUAL OUTPUT**	**RESULT**
Favourite ‘A Dorian’. Select another random scale. Fill scale form with ‘A Dorian’.	The heart icon was only filled when ‘A Dorian’ was selected. When it was not selected, the heart icon was hollow.	✓
EVIDENCE		
		

U11		
TEST INPUT	ACTUAL OUTPUT	RESULT
Favourite 'A Dorian'. Select another random scale. Choose 'A Dorian' from the favourites list.	The scale form was filled out with 'A Dorian'	✓
EVIDENCE		

U12		
TEST INPUT	ACTUAL OUTPUT	RESULT
Click the 'Logout' button.	The page returned to its default state.	✓
EVIDENCE		



Usability Testing

USERS PLAYING THE KEYBOARD (T1)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	The inexperienced user should be able to figure out that clicking a key will play it.	The user took a bit of time but managed to play the keys without any trouble.	This shows that my skeuomorphic keyboard design makes my website intuitive to use, even for an inexperienced user.
Experienced User	The experienced user should not have any issues at all playing the keyboard.	The user figured it out straight away, with no hesitation.	This shows that my interface is usable and the keyboard is completely fit for purpose.

USERS PLAYING A SCALE (T2)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	The inexperienced user may take some time to figure out how to pick a scale but shouldn't need assistance to accomplish the task.	The user took some time to realise that both the starting note and scale type had to be selected. Once they figured this out, they were able to play a scale of their choice with ease.	My webpage may need a 'help' button to make it accessible to everyone, but the average user will find this part of the website easy to use. The scale form is therefore useable.
Experienced User	An experienced user may need to trial and error to fully understand the scale form, but once they have passed that boundary, they should be able to pick and play any scale with ease.	The user was very quick to choose a scale of their liking and did not make any mistakes in the process of playing the scale.	The user interface clearly makes sense to the user. If the user interface is intuitive, it is way more usable and easy to learn.

USERS REGISTERING (T3)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	An inexperienced user should easily be able to spot the register button on the right and notice the register form appear on the screen.	The user quickly clicked on 'Register' and created an account. They paid attention to the password assistance and completed the task without a mistake.	The register process is very clear and easy to use, even on the first attempt.
Experienced User	An experienced user should have no problems registering.	The user created an account without hesitation.	The simplicity of the register process helps all users to register quickly when they desire to. It is good that users do not have to waste time figuring out how to register.

USERS LOGGING ON (T4)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	An inexperienced user will be able to find the log in form with ease. It is clearly labelled, with a light to indicate if it has been pressed.	The user had to attempt logging in twice. The first attempt, they couldn't see their password and had to try again.	Although my site does have a means of showing the password, it is not obvious. Making a 'show password' button would make my website more accessible to less experienced users.
Experienced User	An experienced user should be able to log in without any trouble.	The user logged in; no assistance needed.	The site is usable as a user can return to the website and log in with ease.

USERS FAVOURITING A SCALE (T5)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	An inexperienced user may have to spend some time to learn how the favouriting feature works. They should be able to finish the task on their own.	The user spent some time clicking the button without filling in the scale form. They figured it out eventually and were confident with the feature after learning how to favourite their first scale.	The process could be made more obvious in by disabling the 'Play' and 'Favourite' buttons until the scale form has been filled. Apart from this, the interface mimics the 'Liking' feature of a lot of social media platforms and therefore will be usable to a large demographic of people.
Experienced User	An experienced user may also spend some time learning the system but should become familiar with the system quite quickly.	The user was able to add many scales to their favourite list quickly. They also were familiar with how to unfavourite a scale with no encouragement.	This shows that the 'Liking' systems of existing platforms mean that many people will already be familiar with how to use the 'Favourite' button. The website is therefore usable for a wider population.

USERS LOGGING OFF (T6)			
PERSONA	**EXPECTED RESULT**	**ACTUAL RESULT**	**CONCLUSION**
Inexperienced User	An inexperienced user will be able to find the 'Logout' button with ease.	The user took no time at all to log out of their account.	The account system is intuitive to use.
Experienced User	An experienced user should take no time to log out.	The user went straight for the 'Logout' button.	An experienced user will find the website useable and will not struggle at all to navigate the account system.

Evaluation

This is the final part of my documentation.
I will discuss how fit for purpose, how maintainable and how robust my final product ended up being. Additionally, I will share some of my personal thoughts and feelings on what I have created.

Fitness for Purpose

My program is fit for purpose. Every functional and end-user requirement has been met and comprehensively tested. A user is able to interact with a good-looking, skeuomorphic interface. The user can play a functional keyboard at their own will. If a user wants to, they are able to pick a scale for the keyboard to play back to them. This can be used to learn new scales and sounds or just to practice familiar ones. The user is always able to read the screen interface of the website to see the name of a note or scale. If a user wants to come back to a scale later, they can log in or register. This gives them access to the favouriting feature of the website. They can pick a scale and favourite it to retrieve it quickly. It is easy to identify a favourited scale due to the intuitive heart icon used by many existing interfaces already. Once a user is finished on the website, they can log off at the click of a button. All the specified requirements have been met, and the interface resembles my design very closely for the most intuitive user experience.

Maintainability

My code is maintainable. Every minute process has internal commentary and variable names are consistent and appropriately named to make sure that all my code is understood. I have made sure to include whitespace in between processes and to use indents to demonstrate the hierarchy of the code. This makes my code way more readable and therefore easier to understand. In terms of condensing my code, processes are contained in modules as much as possible. Repeated processes are also put into modules to reduce bloated code. I separated out the PHP files to clean up all the sub-processes into modules. This way I can have a PHP file which contains only low-level code, whilst still functionally executing more complicated processes. I also only wrote one form to perform both the logging in and registering processes. This works by changing the value of the form's submit button and reduces the amount of code in the HTML file. I used CSS selectors to reduce the repetition of similar styling. Overall, I am confident that another software developer could make sense of my code and improve upon it without difficulty.

Evidence of Maintainability

A module of code, with good whitespace, internal commentary, and variable names:

```php
// Function to attempt to register a new user
function registerCheck() {

    // Get database connection
    $conn = $_SESSION["conn"];

    // Get the username and password
    $username = $_SESSION["username"];
    $password = $_SESSION["password"];

    // -------------------------------------------------------
    // AHC : Execute SQL query to insert new data into a database
    // Q2 : To insert a new user into the 'users' table
    // -------------------------------------------------------

    // Run query to add new user to the "users" table
    $query = "INSERT INTO users (username,password) VALUES ('$username','$password')";
    $queryData = mysqli_query($conn,$query) or die (mysqli_error($conn));

    // Log the new user in
    loginCheck();

}
```

The PHP files, split into a file for modules, and a file for readable, low-level code:

phpMain.php

phpModules.php

Robustness

All validation has been tested. All my inputs have validation, but the password creation has the most noticeable validation. This is required by functional requirement F7 ii. and end-user requirement U6. Only the register process is required to validate the password because the password will remain valid once it has been created. It needs to have a letter, a number, a special character and needs to pass a length check to be valid. It is fed back to the user, so they do not make a mistake. Even if they did submit an invalid password, the program is robust and does not crash. From all my testing during both implementation and requirements testing, I can conclude that my code is very robust, and it is most likely impossible for invalid data to cause any errors.

Concluding Thoughts

I am so proud of what I have created. As a musician, creating my own instrument was incredibly fulfilling. I have never built anything of this scale from scratch before and I truly exceeded my own expectations. I am now aware of how much time and care such a creation needs, as well as how important it is to document what you have done. I have dived headfirst into the world of backend code and improved my grasp of front end. There are skills that I have developed that will give me a large advantage in university next year.

If there is anything I would change, or add to my website, I would love to widen the actual musical capability of the instrument. It would be awesome to be able to program in your own sequence of notes to play instead of a few scales; or to change the sound of the instrument. I would also enjoy including more animations to my site to make it flashier and more engaging. Overall, I am incredibly happy with how my website turned out and what it means for what I can create in the future.

I hope you liked my website as much as I enjoyed creating it!