**Saint Augustine’s College, Sydney**

**Software Engineering Year 12: Programming Project**

**Project Documentation: Christopher Pereira**

Table of Contents

[Defining and understanding 3](#_Toc194481484)

[Task Definition 3](#_Toc194481485)

[Legal and ethical considerations 4](#_Toc194481486)

[Functional and non-functional requirements 5](#_Toc194481487)

[Functional Requirements 5](#_Toc194481488)

[Non-functional Requirements 5](#_Toc194481489)

[Implementation Method 5](#_Toc194481490)

[Planning and designing 6](#_Toc194481491)

[Storyboard 6](#_Toc194481492)

[Context Diagram 7](#_Toc194481493)

[Data Flow Diagram 8](#_Toc194481494)

[Structure Chart/Class Diagram 9](#_Toc194481495)

[Algorithms 10](#_Toc194481496)

[Flowchart 10](#_Toc194481497)

[Algorithmic descriptions 10](#_Toc194481498)

[GANNT Chart 11](#_Toc194481499)

[Implementation 12](#_Toc194481500)

[Development log 12](#_Toc194481501)

[Testing 13](#_Toc194481502)

[Test Table 13](#_Toc194481503)

[Project showcase 14](#_Toc194481504)

[GitHub repository 14](#_Toc194481505)

[Evaluation 15](#_Toc194481506)

[Project Reflection 15](#_Toc194481507)

# Defining and understanding

## Task Definition

When I start playing the guitar, it is common practice for me to warm up my fingers and my picking hand. But as I begin to play, I end up finding myself trying to figure out what chords or scales I should play. Moreover, I end up playing the same old boring shapes, even though I am practicing, and I end up not learning anything new and exciting which challenges me.

In this project I intend to create a software application which will have the ability to generate warm-up exercises for guitarists of varying ability. The exercises will contain various chords, progressions, scales and arpeggios which will be drawn from a database containing various patterns. Additionally, users will have the ability to indicate their confidence in particular patterns. This data will then be stored in a database which will be considered in the exercises

Identify any limitations they may encounter during development, such as time constraints, resource availability, or existing technical skills.

Possible limitations in the project will include the fleshing out of ideas and concepts such as the preferences and personal tailored experience in exercises. How will the user indicate their preferences? Will it be stored in ‘.db file’ or by some other means?

## Legal and ethical considerations

During the development of the application the following legal and ethical considerations will be considered…

1. Data Privacy and Security -

The user preference data stored within the app’s database must remain confidential and protected from unauthorized access in order to uphold a user’s privacy and maintain transparency.

To reduce the risk of data breaches or leaks, all user data will be encrypted using a private session key, ensuring that only the individual user can access their information during a session. Additionally, users will have the ability to delete or reset their stored data at any time, giving them greater control over their experience and enabling a more personalized interaction with the app. A clear and accessible privacy policy will also be included to inform users about how their data is collected, used, and protected.

2. Accessibility -

The app should be accessible to guitarists of variable levels of skill and should aim to be compatible with players with varying physical and cognitive abilities (if achievable to put into the app)

Settings with a variety of settings such as high-contrast themes or colourblind settings. Additionally, the display of the app could possibly be changed to account for mobile users

Through these core ideas and suggestions, the app can become more open and accessible to a wider audience of guitar players making a user’s experience more enjoyable and engaging

## Functional and non-functional requirements

Functional Requirements

|  |  |
| --- | --- |
| Requirement | Description |
| Chord Chart Display/Generation | The app will be required to have a function which enables it to take out a specific chord shape from a ‘chord database’ and then displayed it on a chart.  Moreover, the chart should indicate the finger positions, the number fingers to be used in the shape (ranging from 1-4) and which strings should and shouldn’t be played |
| Chord Progression Display/Generation | The app will be required to have a function which enables it to take out a variety of chords from a ‘chord database’ and arrange it in a progression (i.e. A / B / A / Em) |
| Scale & Arpeggio Display/Generation | The app will be required to have a function which enables it to take out a specific a scale or arpeggio from a database, then display it in a guitar [TAB](https://en.wikipedia.org/wiki/Tablature) format |
| Tutorial and Guide | The app will be required to help explain to users and in particular beginner guitarists the basics on how to read tablature ([TAB](https://en.wikipedia.org/wiki/Tablature)). Additionally, it will also explain to users how to read chord progressions and other necessary formats required to progress in the user’s skill.  Videos can also be provided to help further explain and elaborate on concepts which users may find difficult to understand |
| Login System | A login system will be used to enable more customisation and personalising of a user’s exercises.  This will be achieved using a database endemic to a specific account which contains information on preferences and weaknesses detected in a user’s feedback on particular patterns within exercises |

Non-functional Requirements

|  |  |
| --- | --- |
| Requirement | Description |
| Loading & Extraction of shapes/patterns | When the data for the patterns and shapes of specific chords and scales are being generated, there should be a minimal loading time of at least under 0.5 second.  If there loading time is greater than 0.5 or even worst greater than a second this will likely frustrate users and deter them of the app |
| Security and Encryption | The application **will need** to have a form of security or encryption such as ‘SHA256’.  Encryption software can be used to ‘hash’ a user’s session key, which allows for a device and a server to communicate and sent data back and forth between each other. Hashing prevents any 3rd Party from intercepting and possibly compromising transmitted data. |
| Transparency and Ethical Data use | Data transparency is vital to gaining the trust of users. This can be achieved through the use of a privacy policy which clarifies what data is being used by the app.  Additionally, a user should be asked for consent to use the data gathered from their exercises. And users should be able to view the data and either delete or edit the data. |
| Cross-Platform Compatibility | In order to make the application as highly accessible as possible, it would be best to allow for the app to adapt, and change based on the OS/device being used to view the website  This will optimise the experience and satisfaction of the user towards the app |
| Usability | The app should have an intuitive design and aesthetic which is easier to understand, learn and navigate through.  More specifically, buttons should be explicit, and any descriptions given in text such as on the tutorial page should be easy to read and understand.  This will allow for the application to be reliable and easy to use for new users |

Implementation Method

There will be two main implementation methods utilized during the development and release of this project.

1. Phased Implementation – This is when the system is introduced in various stages which may focus on specific and integral components of an application.
2. Pilot Implementation – This is when the new system is trailed by a small group of users.

A combination of these two methods would be very useful in the development of the app because they are reliable in retrieving feedback which is taken from test users, and errors within individual components. This feedback then can be implemented into the website further fuelling its development and constant improvement up until the full release date.

In a serve scenario of a leak, it would be catastrophic, and it could expose any sensitive data which has been inputted could be leaked or any undetected vulnerabilities could be exploited. However, this is very unlikely since the code will be keep confidential and pilot testing will occur in a contained environment

Since the app is intending to expand on an idea which has no clear predecessors, it would be considered best to have a small group of people test out the app and to give feedback. This feedback will be vital for the development of the app and the fleshing out ideas such the personal tailoring of exercises and the display of TABs and chord charts.

As the system is being tested, it will always be run on the latest version of the code immediately after further developments have been made to the code.

# Planning and designing

Storyboard

A screenshot of a computer

AI-generated content may be incorrect.

Context Diagram

A white circle with black text

AI-generated content may be incorrect.

Data Flow Diagram

A black background with white circles and text

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.Structure Chart/Class Diagram

## Algorithms

Flowchart

A black and white diagram

AI-generated content may be incorrect.

In the two-pointed sections are parts of the code which random pick between either a scale exercise or chord exercise

Algorithmic descriptions

**Generate\_Exercises Function:**

1. Parameters: Chords, Scales, Ratings
2. Receive data: Picks out exercises with a higher rating
3. Returns exercise: Provides a routine containing various patterns with exercises the user finds more difficult

**Draw\_Chord/Draw\_Scale Function:**

1. Parameters: Finger\_Positions, Positions
2. Recieves data: Uses an SVG file to create a chart/TAB to draw on the positions of the pattern
3. Returns Chart/TAB: Provides a display of a pattern on either a chart or TAB

**Submit Feedback Function:**

1. Parameters: Rating
2. Recieves data: Places the rating into an SQL Query
3. Returns feedback: Submits feedback to the user feedback database

## 

## GANTT Chart

# Implementation

## Development log

|  |  |
| --- | --- |
| Development Log Entry 1 | |
| Date | [29/4/25] |
| Week Number | [Week 4] |
| Summary of Work Done | Created a basic python file which contains functions allowing for the creation of tablature and chord charts by using an SVG file. The charts and [TAB](https://en.wikipedia.org/wiki/Tablature) are drawn based on data stored within the classes containing the patterns for each shape and the location of the fingers on the fretboard.  The file contains 2 main classes. The first class (Chord Library), is responsible for storing chord shapes, drawing chord charts and generating chord progressions. The second class (Scale Library) is responsible for the storing the patterns scales and arpeggios and displaying those patterns on a [TAB](https://en.wikipedia.org/wiki/Tablature). |
| Challenges and Solutions | Challenges –   * During the development of the foundational parts of the application, it was unclear whether the patterns were to be stored in a large ‘.db file’ or somewhere else * Another concept which was still quite unclear at the time was if the pattern could be replicated across the fret board (i.e. [Barre Chord](https://en.wikipedia.org/wiki/Barre_chord)) would it be best to store the same pattern but in different spots over the fretboard   Solutions –   * An attribute is used to store multiple dictionaries which contain the finger positions for the different chords and patterns.     These dictionaries are then used by the code to create the displays for the [TAB](https://en.wikipedia.org/wiki/Tablature)’s and chord charts   * Furthermore, within the different classes there were various methods (depending on the class) on how to move and use shapes which could be repeated continuously on the fretboard     In the Chord Library, within the dictionaries there is an attribute which stated whether the chord could be ‘barre’ (played across the fretboard). By a chord having this attribute, it would allow for the moveable chord function to be used in order to change the position of the chord.     * A similar system is used in the Scale Library however, it does not use an attribute to indicate the location of the fret, rather, a random root note is chosen for the scale/arpeggio to be based in     After a key is chosen then the code will move the position of the scale in order to account for the difference in location |
| Milestones Achieved | Creating foundational systems for the app including…   * Chord Generation and Storage * Scale and Arpeggio Generation and Storage * Chord Progression Generation |

|  |  |
| --- | --- |
| Development Log Entry 2 | |
| Date | [20/5/25] |
| Week Number | [Week 6] |
| Summary of Work Done | Created a basic Chord and Scale exercise pages which display basic tablature and charts.  Successfully developed and implemented a basic dashboard to allow users to access different exercise types such as Chord Practice, Scale Practice, Daily Exercise and the Tutorial.  Furthermore, I successfully developed a ‘Daily Exercise’ page which takes the various exercises contained within the chord and scale exercises and puts them all together in one large exercise.  As users complete the activities within the exercise, users will be able to give a rating on how confident they felt when participating in the activity. |
| Challenges and Solutions | Challenges –   * During this part of development, there were no adversities, as all the fundamental components had already been coded into ‘chordify.py’, which had saved development time of the other components of the dashboard and especially the different exercise types (daily, scales and chords) |
| Milestones Achieved | Milestones during this period include…   * Implementation of a dashboard to access different exercise types and the tutorial * Development of the different exercise types (daily, scale and chord) which allows for users to either focus on specific types of skills like scales or to focus on all the different types of skills found in basic exercises |

|  |  |
| --- | --- |
| Development Log Entry 3 | |
| Date | [29/6/25] |
| Week Number | [Week 12] |
| Summary of Work Done | Successfully implemented a Feedback System, where once a user has completed an exercise on a particular shape or pattern, they are able to rate how challenging it was to play on scale from 1-10. The data gathered from this is then submitted to the user feedback database.  The ratings of patterns are then compiled to find an average rating. This is then considered by the code during the generation of exercises, so if a pattern (i.e. G Minor Pentatonic) has a high rating such as 8, it is more likely to appear in a newly generated exercise.  Furthermore, additional pages had been implemented to further flesh out the concept the application such as…   * Tutorial Page – Contains a basic overview of the various systems within the app such as on; ‘how to read TAB,’ ‘how to read a chord chart,’ and ‘how to read a chord progression.’ * About Page – Contains a basic overview of how the app was able to personalise and develop exercises, the purpose of the app and a little blurb about the developers. * Settings Page – Contains a table showing all a user’s data. More specifically the average ratings of the feedback given by the users on different chords and scales. This display not only allows for users to see their data but delete their data, this enables the app to maintain a level of transparency over a user’s data. Additionally, the user is able to change their skill level (beginner, intermediate and expert) * Privacy Policy – Reassures users of the proper management and appropriate use of the data gathered from them   Finally, I added a way for the user to choose their skill level. When a user first logs into the app, a pop-up will appear asking for what their current skill level is (beginner, intermediate or expert). By being at a certain skill level it will restrict certain exercise or pattern from appearing in a user’s newly generated exercises. |
| Challenges and Solutions | Challenges –   * Occasionally throughout the development during this time whilst using ‘Github Copliot’ to either implement or change a present feature, when the code was running there would be errors multiple errors encountered   An example of this was when I was editing the feedback database and altered the code in chordify.py for the ‘ScaleLibrary’ the scale exercise was completely broken and met with the following error…     * However, overall, there were very few errors or difficulties encountered in the further development of the feedback system and the addition of other pages such as the about page   Solutions –   * When encountered with the scenarios such as with the Scales Exercise page, usually I would have to either…   Keep querying copilot with more increasingly refined and detailed responses or revert to a previous version of the code then implement and incorporate the features I intended to add at the time |
| Milestones Achieved | Milestones during this period include…   * Implementation of the about, settings, and tutorial pages * Creation of a simple privacy policy * Development and completion of a feedback system which influences the generation of exercises * Added a system to determine whether the user has a skill level which is beginner, intermediate or expert |

# Testing

## Test Table

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test ID | Category | Test Case Description | Input to Provide | Expected Output | Actual Output | Pass/Fail | Action Taken |
| Test 1 | State Testing | Once a new user signs up for a new account, they select their skill level as beginner. If they do this, they will be able to only see a limited range of chords and scales | Press Chord Exercise button | Possible Chords: A (Open), C (Open), etc… | C (Open), F(Barre), A (Open), G (Open) | Fail | Change the logic in the code so during the generation of an exercise the process of selecting certain patterns will also be influenced by what a user’s skill level is |
| Test 2 | State Testing | Once a new user signs up for a new account, they select their skill level as beginner. If they do this, they will be able to only see a slightly broader range of chords and scales than a beginner would | Press Chord Exercise button | Possible Chords: A (2), C (2), etc… | G (Open), F(Barre), A (Open), C (Open) | Pass | N/A |
| Test 3 | Boundary Value | When a user has completed an exercise, they are asked to select on a scale 1-10 how difficult they found the chord/pattern | Press Next Exercise Button  Select 10 on the scale  Press submit feedback Button | Feedback added to ‘Feedback.db’  Next Exercise | Feedback added to ‘Feedback.db’  Next Exercise | Pass | N/A |
| Test 4 | Boundary Value | When a user has completed an exercise, they are asked to select on a scale 1-10 how difficult they found the chord/pattern | Press Next Exercise Button  Select 1 on the scale  Press submit feedback Button | Feedback added to ‘Feedback.db’  Next Exercise | Feedback added to ‘Feedback.db’  Next Exercise | Pass | N/A |
| Test 5 | Path Coverage | Once a new user signs up for a new account, they select their skill level as beginner. They will be able to have full access to all of the different exercises with no issues | Press the Daily Exercise Button  Complete Daily Exercise  Press the Scales and Arpeggios Exercise Button  Complete Scales and Arpeggios  Press the Chord Practice Button  Complete Chord Practice | All patterns should be loaded without any issue or bugs | All patterns loaded without any issue or bugs | Pass | N/A |
| Test 6 | Path Coverage | Once a new user signs up for a new account, they select their skill level as intermediate. They will be able to have full access to all of the different exercises with no issues | Press the Daily Exercise Button  Complete Daily Exercise  Press the Scales and Arpeggios Exercise Button  Complete Scales and Arpeggios  Press the Chord Practice Button  Complete Chord Practice  N/A | All patterns should be loaded without any issue or bugs | All patterns loaded without any issue or bugs | Pass | N/A |
| Test 7 | Path Coverage | Once a new user signs up for a new account, they select their skill level as expert. They will be able to have full access to all of the different exercises with no issues | Press the Daily Exercise Button  Complete Daily Exercise  Press the Scales and Arpeggios Exercise Button  Complete Scales and Arpeggios  Press the Chord Practice Button  Complete Chord Practice | All patterns should be loaded without any issue or bugs | All patterns loaded without any issue or bugs | Pass | N/A |
| Test 8 | Path Coverage | In the settings of a user’s interface, they are able to delete the data gathered on them | Press Delete Button for data on A chords  Pop up – ‘Delete all feedback for this chord shape?’  Press Ok button | All A chord Data Deleted | ValueError: invalid literal for int() with base 10: 'None' | Fail | Located the source of the error and tried to figure out the why the source was giving off the particular error |
| Test 9 | Path Coverage | In the settings of a user’s interface, they are able to easily change the skill level their account is set at | Beginner skill  Press on drop down menu  Select Expert skill | Skill level set to Expert Skill | Skill level set to Expert Skill | Pass | N/A |

# Project showcase

GitHub repository

|  |  |
| --- | --- |
| Heading | Detail |
| Description | **What is our Goal?**  Our main goal is to create an application which can generate basic warmup exercises for guitarists of varying levels!  This can be a player which is either just starting out, or someone who is more experienced and might be struggling to figure out the best way to warm up before their usual practice routine.  We provide users with a basic routine which cover scales, arpeggios and chords, allowing for users to expand on simple guitar theory and technical skills which is vital in understanding the guitar |
| Installation Instructions | **Installation Guide**  **If you are interested in trying out the app please follow the installation instructions below...**   1. Go to the main page 2. Press the green button with code<> on it 3. Then go to GitHub CLI 4. Copy the URL 5. Then use your Software Application (make sure your logged in your Git Account) to clone the repo   If you have any issues, you can also download this file as a .zip and use your Software Application (VSC) to access the file. |
| How to Use | How to Use?   1. Sign up for an account if you don’t have one already 2. Login, upon doing so you’ll be greeted by a pop up which asks for your skill level.   Please select one of the three options presented as they help to give a better idea of what type of patterns an exercise should be given   1. Now all you have to do is click either Daily, Scale exercise or Chord practice and you will have access to a randomly generate exercise with different patterns 2. Once you are ready to move onto the next pattern or exercise press the ‘Next Exercise’ button. Afterwards you will need to fill out a feedback forum by selecting how challenging you found playing the exercise on a scale from 1-10 3. Once you have submitted your feedback you can proceed with the next exercise. Repeat this same process until the entire routine has been complete 4. If you are interested in viewing all the feedback collect or removing some of it please visit the home page and go to the settings page |
| License Information | Mention the type of license the project is under. |
| Visuals | Include screenshots or graphics that make the README.md visually appealing and give a quick idea of what the project is about. |
| Acknowledgements | Acknowledgements  I would like to acknowledge my teacher Mr Fong and my friend Jonah who helped to provide critical feedback which helped to stimulate the development of this project |
| Author Details | **About Developer**  I have played the guitar for many years, and I have time and time again always encountered the same problems when trying to warm up for my guitar practices. I struggle to figure what I exactly wanted to do, and many other guitarists also have this same problem  So, I decided to develop this application for my Year 12 Software Engineering Major Work as it covers a field, I’m greatly interested in. |
| Clean Directory Structure | Organize your files in a logical manner, with folders for different types of files if necessary (e.g., separate folders for source code, assets, and documentation). |
| Documentation | A link to your .docx documentation file for easy access. |
| Additional Details | None |

# A person holding a guitar and a clock AI-generated content may be incorrect.

Link: [imamchris/Guitar-Warmup-Coach](https://github.com/imamchris/Guitar-Warmup-Coach)

# Evaluation

Project Reflection

The main aim of this project in the end was to create an application which would be able to generate basic warmup exercises for users of varying levels of ability and capability on the guitar.

More specifically the main components of the website I intended to develop within the website were…

* Chord Progression Display/Generation – The website would extract chords from a chord database. Afterwards it would draw the finger positions of the chord onto a chart so users will be able to interpret then play a chord
* Scale & Arpeggio Display/Generation – The website would extract scale and arpeggio patterns from a database containing patterns. Afterwards it would draw the finger positions of the scale/arpeggio onto a TAB so users will be able to interpret then play
* Tutorial and Guide – The website will also provide a basic tutorial to help inexperienced users to better understand the various systems being used on the website. Such as the Tablature, chord charts and progressions
* Login System – A system which will allow for users to have separate accounts which allows for users to have a more personalised experience based on their feedback they have submitted for exercises

Majority of the project was successful, such as with the development of all of the functional requirements and the non-functional requirements (loading/extraction of patterns, hashing and security measurements of the website, appropriate use of data and transparency of data).

One of the most challenging parts of the development in specifically the functional requirements was trying to find the appropriate library for displaying chord charts, progressions and scale TABs. There were a wide range of possible options which I could’ve used. However, none of these options suited my requirements and preferences. So, to find a system which accommodates for the needs required for the foundational components of my application, I ended up creating my own specialized python file called ‘chordify’.

Chordify contains two main classes, which include…

1. Chord Library – Stores a set of dictionaries containing data for chord patterns such as finger positions and the location on the fretboard
2. Scale Library - Stores a set of dictionaries containing data for the positions and locations of scales and arpeggios on the fretboard

Both classes contain functions responsible for drawing these positions out onto a display. This is achieved by using an SVG file, where functions such as draw\_chord() draw out a chart onto the SVG then places dots onto the chart which represents the finger positions on the fretboard (a similar system was used in Scale Library except the positions are drawn onto a TAB).

After the development of chordify, it saved time in the development of the flask routes and respective html pages. However, the development of these pages heavily relied on the development of AI. This meant that at times I didn’t understand sections of the code and I only had a rough idea of what was happening within the code.

In future projects, I hope to possibly not rely as heavily on A.I in the creation of the code. Another thing which I would like to improve on is to start with the theory of the code first, so that means I have more time to implement solutions to errors in the code.

Overall, this project was a great success and majority of it ran perfectly