## Problem

My client has a Rubik’s Cube and would like an application that can show them the steps needed to solve it from any scrambled state, and help them to learn how to solve the puzzle.

## My Client

My client is a teenage girl who wants to learn how to solve the Rubik’s cube, and wants a program that can show her the steps needed to solve a scrambled cube.

## Client Interview Response

1. **How well do you know how to solve a Rubik’s Cube (1 = No knowledge -> 4 = full knowledge)**

|  |  |  |  |
| --- | --- | --- | --- |
| **1** | **2** | **3** | **4** |
| x |  |  |  |

1. **Which method would you prefer for outputting the instructions?**

|  |  |  |
| --- | --- | --- |
| **Rubik’s Notation** | **3D output** | **Both** |
|  |  | x |

1. **Would you prefer an application that teaches you how to solve the cube alongside showing the steps?**

|  |  |
| --- | --- |
| **Yes** | **No** |
| x |  |

1. **Which cube solving method would you prefer the application to use?**

|  |  |
| --- | --- |
| **Beginner Method** | **CFOP** |
| *A method which solves the cube in layers and involves only a low number of algorithms. It is therefore is one of the easiest methods to learn.* | *Cross, First 2 Layers, Orientate Last Layer, Permute Last Layer (CFOP) is an advanced layer-by-layer method which employs the use of selecting from many algorithms to solve each layer in fewer moves than the Beginner’s Method. It is an advanced method used for speedcubing.* |
| x |  |

1. **Are you likely to stop mid-way through solving, and want to save the current stage of the cube?**

|  |  |
| --- | --- |
| **Yes** | **No** |
| x |  |

1. **Are you likely to want to save/load multiple cubes?**

|  |  |
| --- | --- |
| **Yes** | **No** |
|  | x |

1. **Do you have any comments/ideas about how you would like the state of the cube to be input?**By clicking in the colours on screen – not having to type them. Ideally by a 3d diagram which you click on.
2. **Do you have any comments/ideas about how you would like the instructions to be output?**3D animation with step by step click through
3. **Do you have any comments/ideas about the saving/loading of cubes?**No
4. **Do you have any other comments/suggestions for the application?**No

## Conclusion from questionnaire

|  |  |
| --- | --- |
| Question No. | Conclusion |
| 1 | The client has no knowledge of solving a Rubik’s cube, the application should therefore go through the instructions 1 step at a time. It should also follow the beginner’s method, and should output the steps of the beginner’s method alongside the instructions to solve the cube. |
| 2 | The client wants a 3D animation showing how to solve the cube, alongside the Rubik’s notation for that step. |
| 3 | The client wants the application to teach her how to solve the cube, as well as showing how, so it should output the steps of the beginner’s method algorithm. |
| 4 | The client wants the application to follow the beginner’s method algorithm. |
| 5 | The client wants to be able to save the state of the cube mid-solving. |
| 6 | The client only needs to be able to load one cube at a time, so when saving cubes the program can overwrite other cube files. |
| 7 | The client wants to be able to input the state of their cube via clicking – the system should use a GUI. Ideally a 3d diagram, although this may have to be an optional, extension objective. |
| 8 | The client wants the instructions outputted via 3D animations, one step at a time. |

## Objectives

After a discussion with my client we have agreed upon the following set of objectives for my project. The system should:

1. **Allow the user to input the state of their cube**
   1. Use a GUI to enable easier input
   2. Allow choice from any of the 6 colours
   3. Allow each sticker to be edited after it has been entered
   4. Allow for input via a 3D GUI *(optional extension objective)*
2. **Have a menu system**
   1. Should be able to return to previous screens
      1. Data should be saved or a confirmation message should be displayed
   2. The menu system should be easy to navigate
      1. Clearly labelled buttons
      2. Similar layout and style throughout the solution
3. **Validate if the entered cube is a valid cube**
   1. Check there are 9 of each colour inputted
   2. Check that no 2 opposite colour stickers are on the same block
   3. Check that there are no of the same colour stickers on the same block
4. **Work out the instructions needed to solve the cube from its inputted state**
   1. The instructions should solve the cube fully
   2. The instructions should be fully simplified in terms of consecutive rotations of the same face
      1. 2 quarter turns in a row should be replaced with a half turn
      2. 2 quarter turns in opposite directions in a row should both be removed
      3. 2 half turns in in a row should both be removed
      4. A half turn and quarter turn in a row should be replaced with the opposite quarter turn (e.g. ¼ clockwise + ½ 🡪 ¼ anticlockwise)
   3. The instructions should follow the Beginner’s Method
   4. The instructions should be able to solve the cube from a half-solved state without restarting (e.g. if the green face is correctly solved it should carry on from that point)
5. **Output the instructions to solve the cube**
   1. Output each step one at a time
   2. Allow the user to replay steps
      1. Steps should be able to be undone
      2. Steps should be able to be repeated once undone
      3. Steps should be able to be undone and replayed any number of times
   3. Output the instructions in a 3D graphical format
      1. If 3D is not available on the user’s machine, output a suitable message and switch to text-based instructions
      2. Display the cube turning to clearly show the user the instruction
      3. Allow the user to control the speed of rotation
   4. Also offer the option of text-based instructions using Rubik’s Notation (see **Appendix B**)
6. **Allow the user to store the cube in the middle of being solved**
   1. Save the cube in a file
   2. Allow cubes to be retrieved from the files at the same instruction that the user saved it on
      1. Store the current instruction alongside the cube
   3. Allow cubes to be loaded in from files
      1. If the current instruction file is deleted then output a suitable message to the user

As this is a challenging and time-consuming problem I have discussed with my client the possibility that, as with any project, there may not be time to add all of the requested features. Through discussion, we have decided that objective 1.4 is non-critical and this will be added last, or not added at all, depending on the time left once all of the critical features have been added. To accommodate this, and any problems that may arise during development, I will use an agile development method throughout the project.