ELEC2660 – Mobile Applications

Documentation – Sudoku App

Contents

[Specification 2](#_Toc498693059)

[Features 2](#_Toc498693060)

[Potential Features 2](#_Toc498693061)

[GUI 3](#_Toc498693062)

[Overall Story Board 3](#_Toc498693063)

[Menu 4](#_Toc498693064)

[Game Screen 5](#_Toc498693065)

[Game Winning Screen 7](#_Toc498693066)

[Solver Screen 8](#_Toc498693067)

[Solved Solution 9](#_Toc498693068)

[Class Diagram 10](file:///C:\Users\Daniel\Documents\University\ELEC2660%20App%20Documentation.docx#_Toc498693069)

[Interaction Diagram 11](#_Toc498693070)

# Specification

The application provides the user with either an interactive sudoku puzzle or a sudoku solver, chosen from the menu. It will provide help for struggling users. It will check all entered values for contradictions in the game’s rules.

## Features

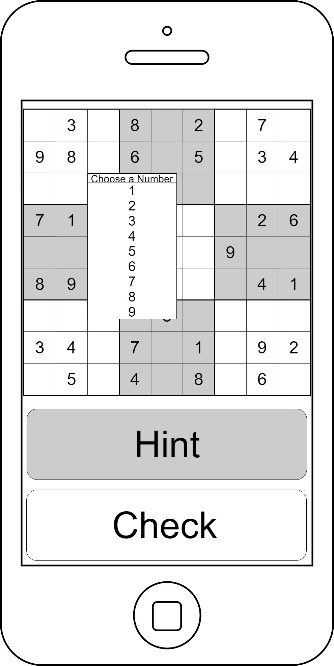
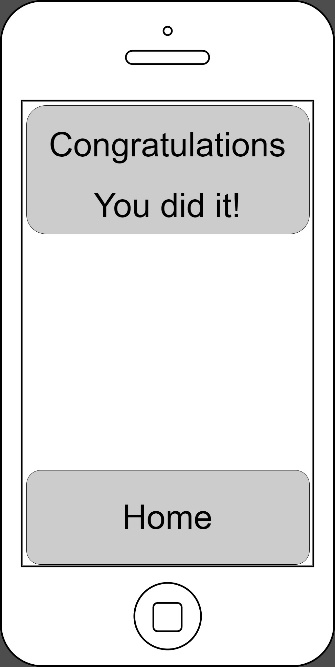
1. Generate a sudoku puzzle for the user to solve
2. Varying difficulties for game
3. It will check each entry to check entries follow the game rules, as the number is inputted (will not notify if wrong, only if rules are broken)
4. Can provide hints to aid in the completion upon request
5. Can check if entries are correct and will clear single boxes
6. Can provide solved solution to incomplete Sudoku Puzzle
7. Will enter values into chosen squares using drop down UI
8. Work in Portrait

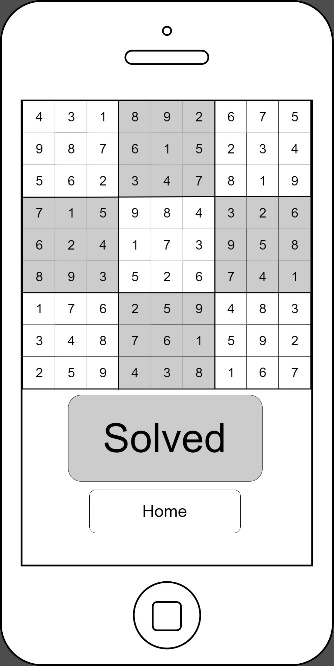
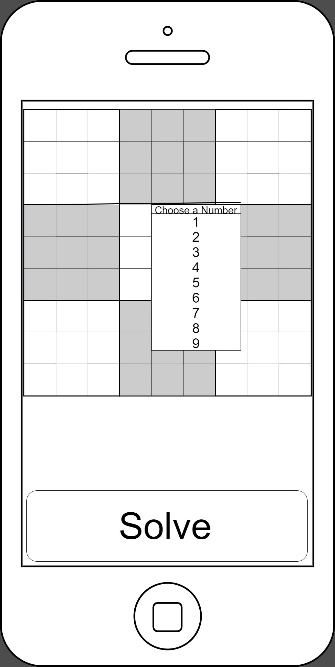
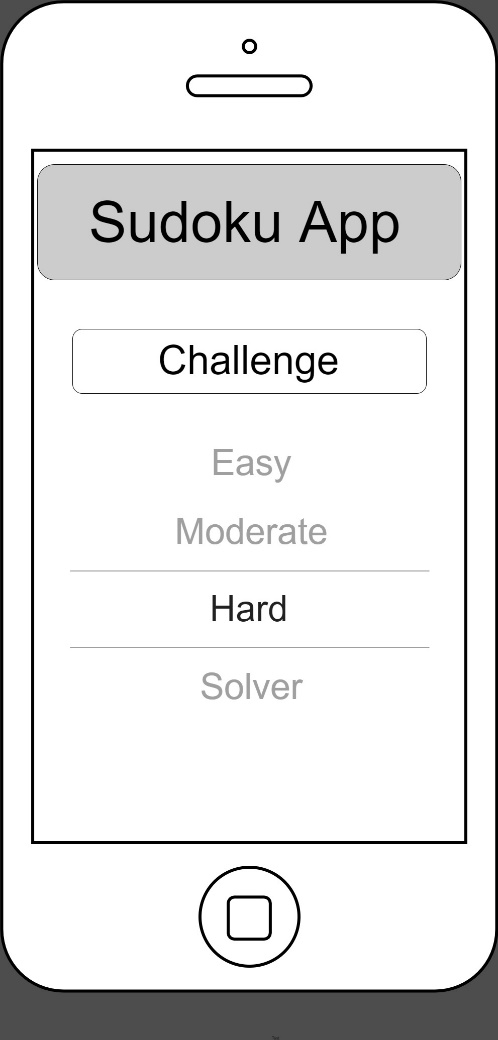
## Potential Features

1. Saving/Loading
2. Timer/Hint Penalty
3. Completion Reward System
4. Step by Step Completion (next solvable square, so not random)
5. Have limit for how many squares revealed per 3x3 grid
6. Work in Landscape as well as portrait

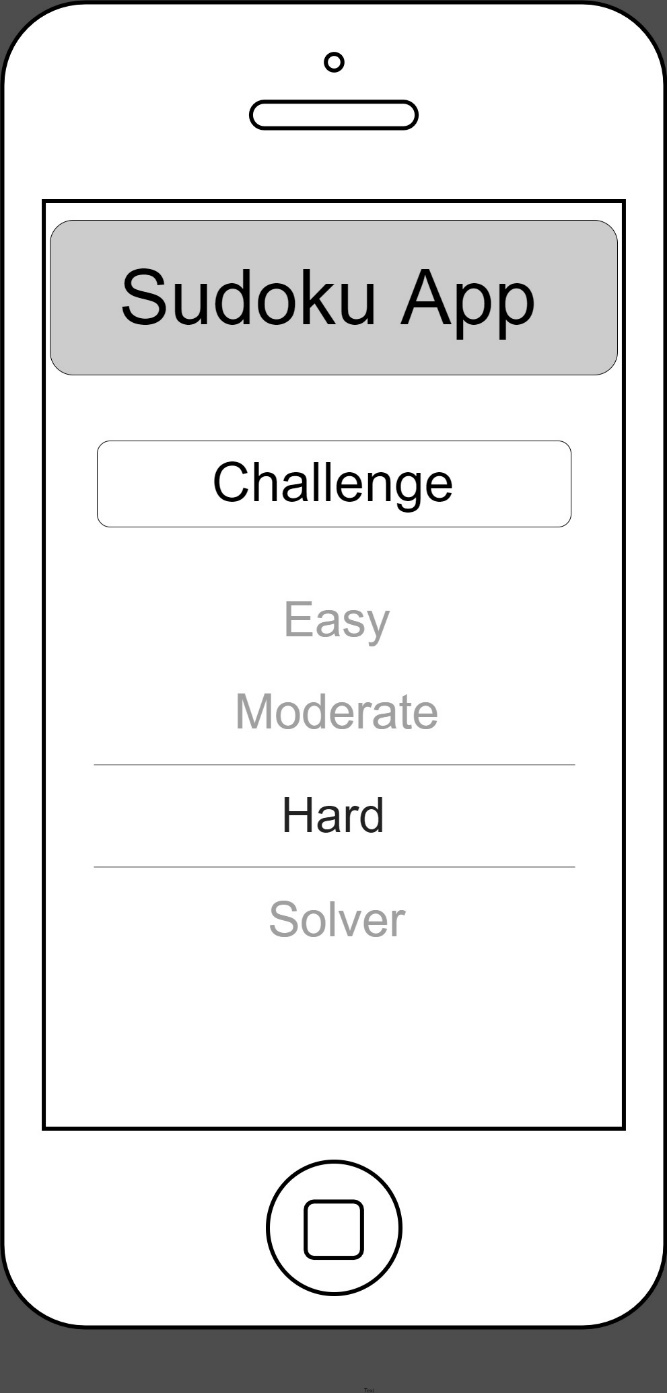
# GUI

## Overall Story Board

­­­­­­­­­



## Menu



**IBOutlet**

**Label**

**IBOutlet**

**Label**

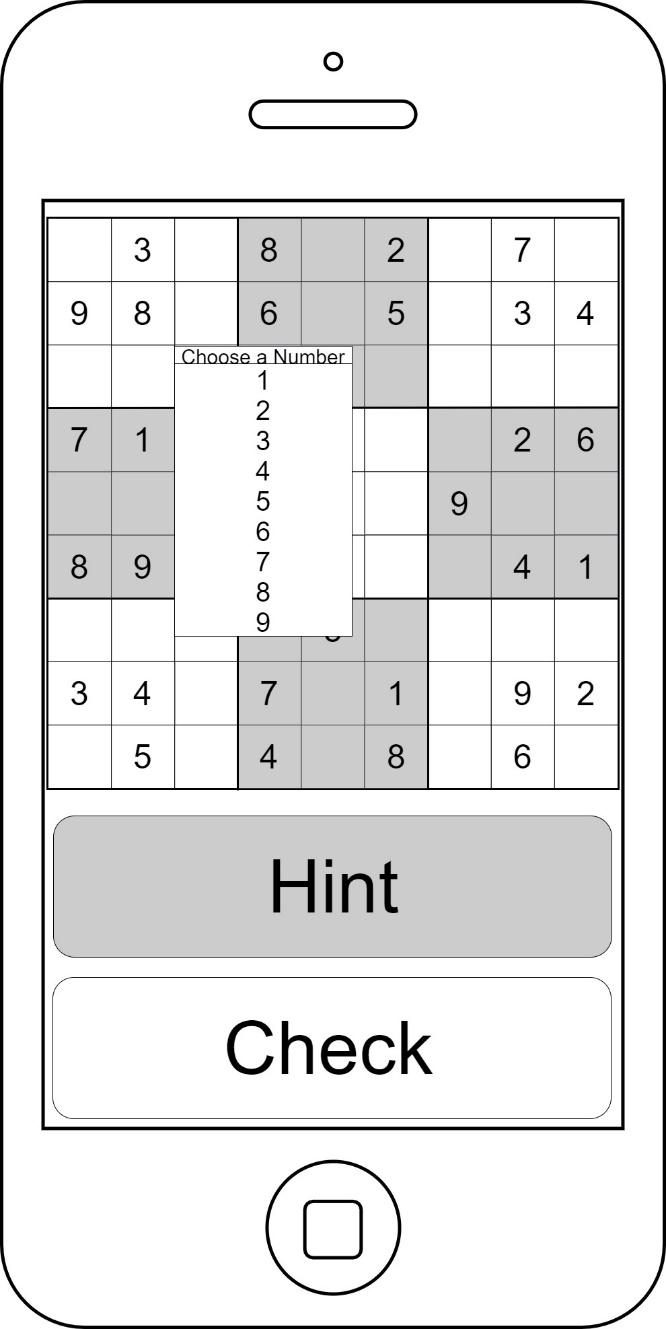
**IBAction**

**UIPicker**

Home screen of app, here the user uses the above picker to choose which game mode they want to use. As well as difficulty for the randomly generated sudoku puzzle, achieved by setting the random sudoku generator to reveal a predetermined number of squares (all random positions). For example, Easy will reveal 40 squares, Moderate will reveal 30 and Hard will show 20 of the 81 squares. A Sudoku Solver is also available by selecting it in the picker.

The colour scheme was chosen to be as monochromatic as it is as it is an attractive colour scheme, as well as suiting the sudoku puzzle style. Ideally it would look minimalistic to suit the plain game style of sudoku as to not be distracting to the user. The game buttons were not chosen with any intent as on a menu screen there are several options to choose from. Therefore, the index finger is most useful on this page. However, for the rest of the pages the key buttons were positioned towards the bottom of the screen for easy access. The grid itself is at the top of the page, once again forcing the index finger use, however this is useful as the user must be accurate to not cause frustration with the user.

## Game Screen



**1**

**2**

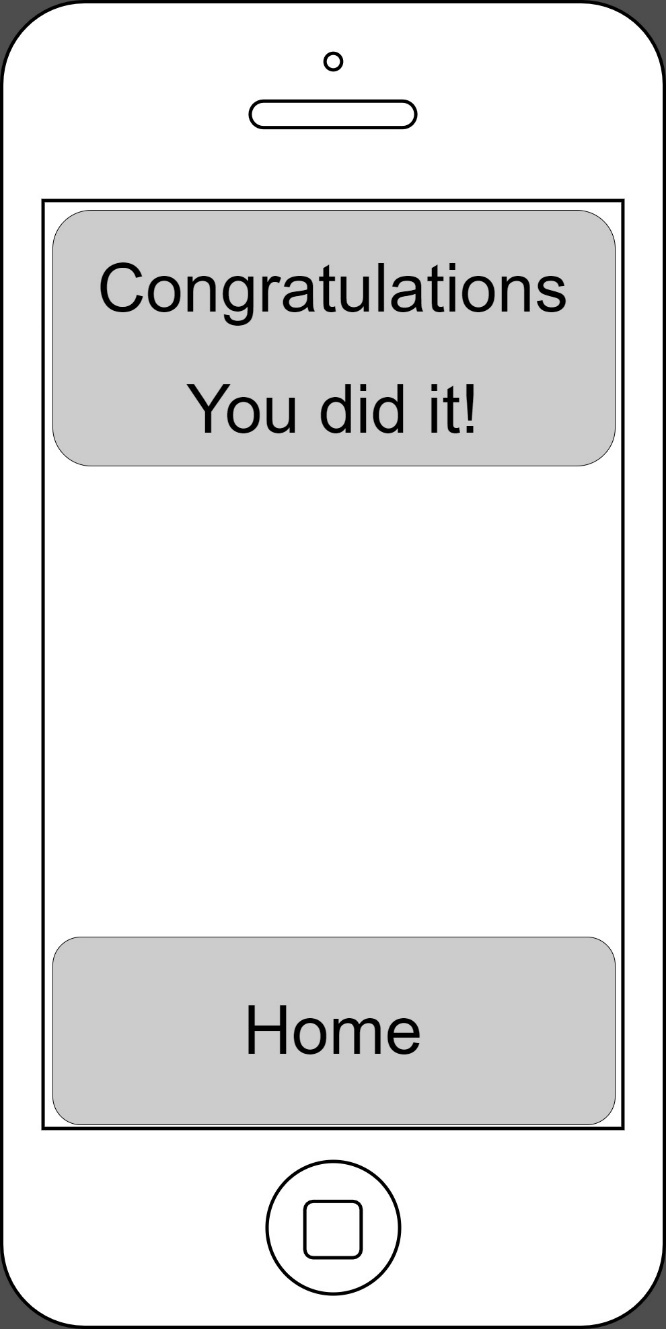
This is the screen shown during the game, that has been created and solved by the app already, using a large if loop to check it is solvable.

1. Sudoku Grid itself is made up of 81 buttons, that will each contain an array of 1 to 9, which will be reduced by the algorithm until only one number goes into each square, and solved the human way. By referencing row, column and the 3x3 square relative to each square to reduce the array. The value chosen to insert into each square is chosen using a drop-down list as shown.
   1. The hint button will, currently, not provide a step by step, explained solution. It will instead reveal one random empty square, this is because of my limits as a programmer and the time constraints to try and complete the very difficult task that would be to produce steps with accompanying reasons. However, if I have time I will
   2. The check button will check all values in the grid so far, cross reference them with the already solved solution and circle all wrong values in red circles.

An algorithm will check each input to check for any entries that conflict with the rules of the game, and flag up any errors. Such as the same number being inputted into the same row/column, the new one will be circled in red. Red was used as an alert and stands out from the grey colour scheme.

Once all values are in the grid, and checked to be correct, the app will move onto the next page.

## Game Winning Screen



**IBOutlet**

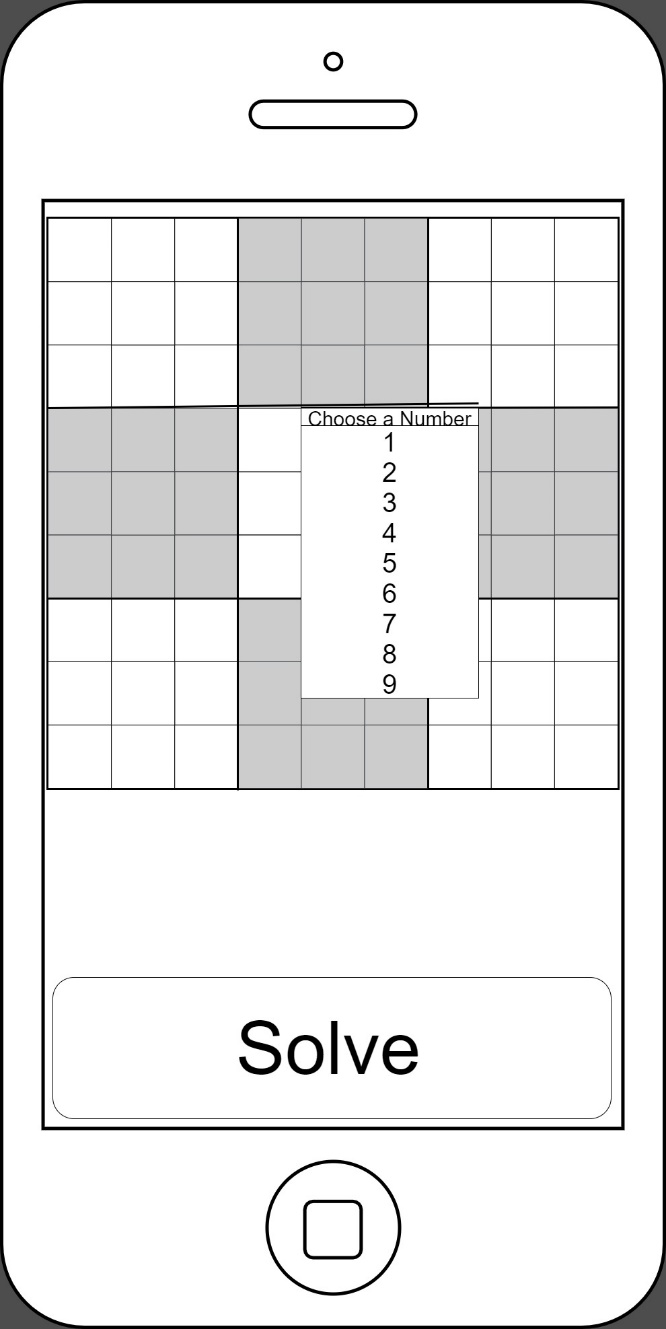
**Label**

**IBAction**

**Button**

This screen is a simple one, simply a button that will transition back to the home screen.

## Solver Screen



**IBAction**

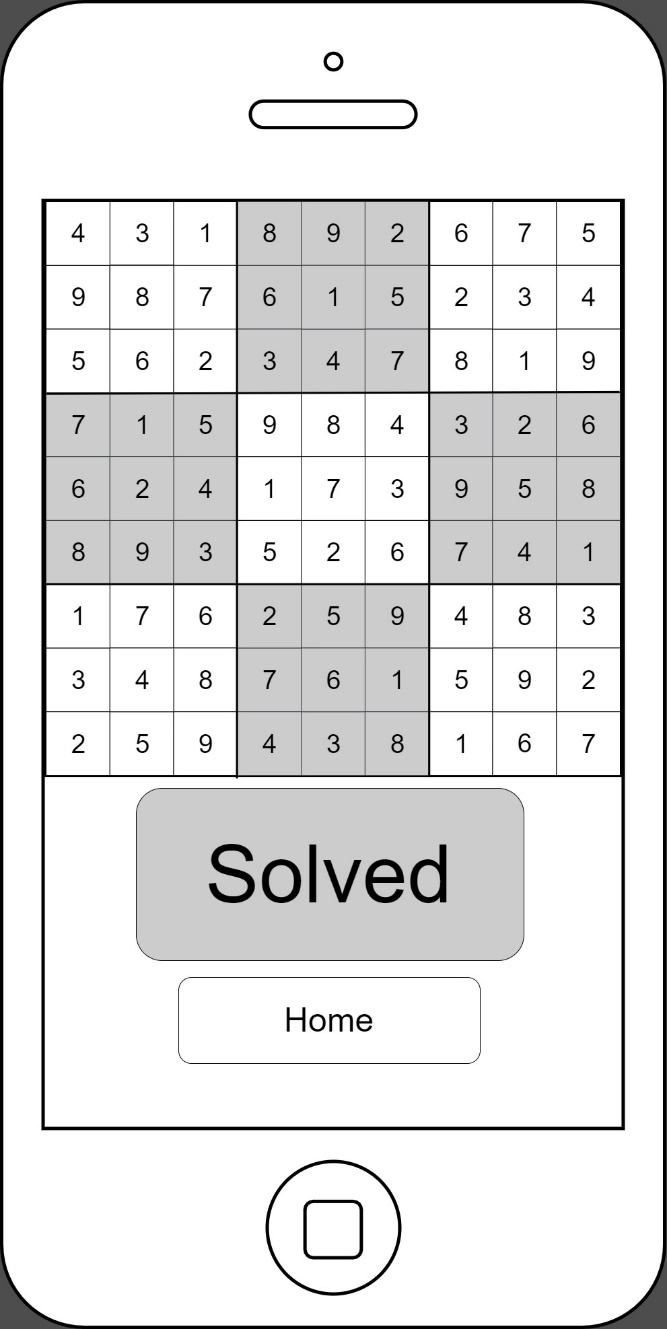
**Buttons**

**IBAction**

**Button**

This page is accessed from using the home page. The user fills in all known values of the sudoku, same way as during game, and by clicking the “Solve” button. Once this is pressed the algorithm will determine and display the solved solution on the next page.

## Solved Solution



**IBOutlet**

**Labels**

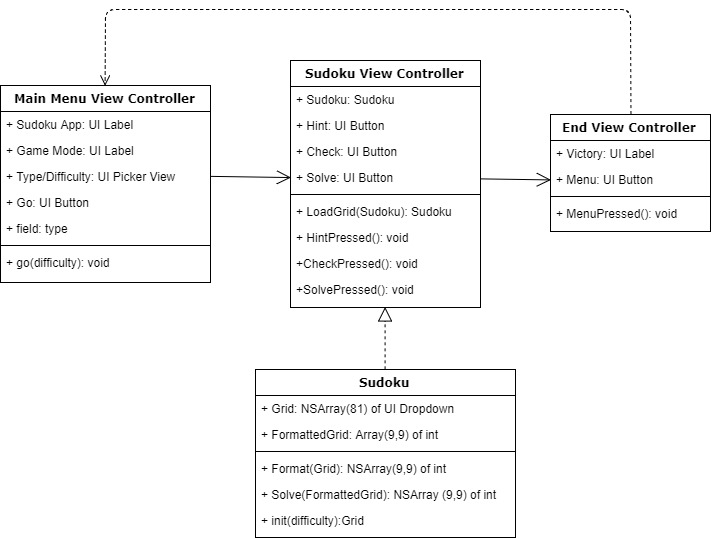
**IBOutlet**

**Label**

**IBAction**

**Button**

This page is also a simple one, just displays the solved sudoku grid. The sole action for the page is the transition back to the home page to start the app again.



# Class Diagram

# Interaction Diagram

