Slide 1 (My Part)
How the puzzle is defined?
How a piece is defined?
How the pieces interact with each other?
How the puzzle is solved?
Displaying the pieces correctly on the screen.
How pieces are picked up and placed back down?

Slide 2 (Background - Jigsaw Puzzle Games)
There are a lot of puzzle games out and about already.
They are limited to orthographic projection, meaning the puzzle pieces are flat on the screen and the user looks top down to all the pieces, there is no sense of depth, as all the pieces are the same size.
Are limited in where the pieces can go and how the puzzle can be solved. Most games use a grid to place the pieces which is boring.
Want to avoid these limitation in the project, want a perspective projection and no grid for the board.

Slide 3 (Background - Jigsaw Puzzle Games)
Picture one shows the grid that the puzzle pieces have to be placed in to solve the puzzle.
Both pictures show the orthographic projection where you see the puzzle from top down.

Slide 4 (Background - Augmented Reality)
The whole idea of the project is to have a virtual puzzle being displayed on a real world board. This is augmented reality.
The idea of augmented reality is to have virtual objects being superimposed onto the real world.
There are a couple games that have done augmented reality and I have listed a couple here.

Slide 5 (Background - Augmented Reality)
Both games are being played on a table using marker tracker. The first shows what the game looks like through the screen. The second shows the game being played through the iPhone.

Slide 6 (Progress so Far)
I have done the game logic, rendering of square pieces with textures, pieces are able to be picked up and placed back on the board, pieces can be rotated, pieces are able to snap together, and as well as the overall puzzle being check to see if it has been solved.

Slide 7 (Game Logic)
First thing was to define what a puzzle piece is;
Each piece has an unique ID, its location on the board given by an x,y coordinate as well as a rotation. All the pieces share the same side length.
Each piece has four edges which may or may not have a corresponding neighbour piece.
Each edge can be in one of three states
Closed meaning it has found and joined up to its neighbouring piece
Open meaning it has not yet joined it neighbour

Invalid meaning that it has no neighbour, for now invalid edges are for the edge pieces of the puzzle, also known as the outside pieces.

Slide 8 (Game Logic)
The Jigsaw puzzle has a number of pieces which is defined by a number of rows and columns. On startup each piece is generated and randomly placed on the board.
The board to place the pieces on has length and width in the x and y directions, as a piece has an x and y coordinate they can be placed within the bounds of the board.
A player can only ever hold one pieces at a time, this stops the issue of a player running around the board picking up all the pieces, and stealing them away from everyone else in the game.

Slide 9 (Interface)
How you interact with the game.
To pick up a piece you currently tap the piece on the screen. The piece is placed in the player's inventory which is shown on the bottom left of the screen. The piece on the board is removed to show that is no longer able to be picked up by anyone.
Currently to place a piece back on the board the player taps on the screen they wish to place the piece, the piece is removed from their inventory and the piece is placed back on the board.
To place a piece in future, we want to have a transparent puzzle piece in the centre of the iPad screen which casts a shadow onto the board, showing that when the player taps the screen to place a piece, it will appear exactly where the shadow is cast.

Slide 10 (Neighbouring Pieces)
As I said before most existing puzzle games are limited to how the puzzle can be solved, due to the pieces having to be placed on a grid. I wanted to avoid that with the project, so the pieces can be placed anywhere on the board, and so the puzzle can be solved anywhere, be it in the centre or the far corners of the board. To do this I had to implement a neighbouring system, which means each piece knows who each of its neighbours are. So for the puzzle to be solved, each piece needs to join up to its neighbouring pieces.

Slide 11 (Rendering the Puzzle)
A piece's parameters determine where on the board they appear, their x and y coordinates as well as their rotation which affects the orientation of the piece. To start with I originally had blank square polygons as pieces for initial testing of the game logic. Now they have textures on them. In future I want to have more interesting polygons for the pieces where the edges are curved to look more like puzzle pieces that should join together.

Slide 12 (Snapping Pieces)
Since pieces can move around the board now, I wanted to implement a snapping feature, which means that when two pieces were meant to join together if they were in a specified range of each other. So a piece being placed on the board that was in range of its neighbour would be positioned to be right next to it. The snapping feature only works on neighbouring pieces, but it doesn't stop the user from positioning two non-neighbouring pieces next to each other.