# Sprint Review 2

Features implemented:

Note: We can include stuff like the architecture planning here

* Created a graphic of our system’s architecture.
* Made many use cases with a few models to go along with them.
* Die Roll feature was pushed to the github
* Game data structure was pushed to

Issues fixed:

* Fixed NFRs and FRs in the SRS
* Fixed user stories that needed to be fixed

Implementation review *(What went well in the implementation, what problems occurred, how problems were solved)*

Went well:

Getting a solid understanding of the underlying structure of our project

Planning our moves bit by bit

Implementing tasks as part of our goal

Problems:

Initial confusion on purposes of some of our architecture

Solution:

Building the architecture design document let the whole team get on the same page as to how all the moving pieces connected with one another. From there,

Changes made:

Plans for next sprint: *(What will be done for the next sprint)*

The primary point of focus for Team Squared’s next sprint will be focusing on and clarifying the *attack plan* outlined earlier in this week of development. From this point forward in our development, Team Squared will be adopting an attack plan which outlines critical functions and implementations of website features that

* *In Python*, create a game list describing, among other things, the number of players, the dimensions of the game’s board, and whose turn it is in the game that the list describes.
* *In Python*, make a gameCreate function which generates a five-digit number to be used as a game’s ID, then adds players, chooses one of the players to go first, and initializes the game board.
* *In ReactJS*, make a query get sent to the API once a player clicks a square on the game board (i.e., /move?square=a2).
* *In ReactJS*, when it becomes a player’s turn, query /dice to generate a six-digit number, representing a dice roll.
* *In Python*, create a dice endpoint and have it return a number between 1 and 6.
* *In Python*, create a function that dissects a query string.
* *In Python*, create a move endpoint, which uses the aforementioned query string reader function to send a player’s move to validateMove(player,square).
* *In Python,* create a validateMove() function which checks a dice roll, then computes if the clicked square is valid. If the square is valid, the game list should be updated to reflect the player’s new location, change the turn to the next player’s turn, calculate a new dice roll, and return True. Otherwise, return False.
* *In ReactJS*, make a generic error message for when /move returns False.
* *In ReactJS*, reflect a player’s valid move on the game board if /move returns True.

Scrum review: *(What went well in Scrum, what could be improved, and what changes will be made)*

In our scrum, we decided to put an attack plan into effect and will be updated every time that we meet. This includes all of the development tasks that we need to get done as first priority. We also discussed how we were going to display our architecture in a professional manner. There was discussion about how everything was going to tie in together and it led to everybody being on the same page in the end. We could improve on being crystal clear with documentation on who is responsible for what parts of the attack plan. I wouldn’t change much from our last meeting as long as we stick to the attack plan.