**Project Name**: ARiderPokemon

**Group Name**: ARider

**Members**:

* Aakash Singh Rathore
* Yash Yadav
* Chriz Januzik
* Sean Conyne

**Description**: The game features a pokemon safari adventure game, in which there are 3 win condition:

* Catch the Legendary Mewtwo: it can be present anywhere in the map. If you’re able to catch it, YOU WIN!
* Catch 10 Pokemons: Catch 10 Pokemons and you win
* See how many Pokemons you could catch

The game showcases 2 maps, a meadow map and a desert Map. Each map has different pokemons, depending upon the pokemon kind. You could encounter these pokemons in the map. The pokemons have been divided into 3 categories, common, uncommon and rare. It is easy to encounter a common pokemon, a bit harder to encounter an uncommon one, and even harder to encounter a rare one. When a pokemon is encountered, the battle screen is displayed and the player can either run, throw a bait, throw a rock, or throw a safari ball to catch the pokemon. Catching a pokemon is sheer luck. However, it is easier to catch a common pokemon, harder to catch an uncommon one and even harder to catch a rare one. The player has to catch the pokemon in 5 turns, but it is possible that the pokemon might run earlier. The player is given 500 steps and 30 safari balls. He loses if he’s out pf steps and hasn’t satisfied the win condition. Besides this, the file menu displays the controls, inventory, pokedex etc. Also, to break rocks the character needs rock smash ability and to surf on water he requires surf ability. Both of them can be found on the map

**Design Patterns**

1. **MVC**: The entire program uses the MVC design in order to keep information hidden from and displayed to the player separate. All calculations and generation are done within the model and are hidden, while the information displayed to the player is in the view within panels, and the controller is what allows the player to interact inside the game.
2. **Builder**: Used by the main controller to handle creation of different elements in the game, such as the Pokemon and map. The main controller class in instantiated in the main of AriderPokemonController class and a window pops up. The creation of the instance of the controller class justifies the use of builder pattern. We just need to create the instance of the controller class and everything is build up and presented to the user as a JFrame
3. **Factory**: Used by the panels to quickly construct the required the windows used for gameplay, such as the map, the battle screen,and the summary screen. We just need to call the panel class and it does everything on its own and displays the output
4. **Observer**: This pattern is used in order to keep track of and display any changes that take place during gameplay. The graphic panel is the observer, while map, Pokemon, and battle are observable.
5. **Iterator**: We did not use the iterator pattern for creating Pokemon or maps, because we found that each of these elements had too many specific characteristics that would make it inefficient to try and create them all the same way.
6. **Singleton**: We did not use the singleton pattern for creating Pokemon because we wanted them to be able to be created freely. Singleton would have created unique instances, when we wanted to use general instances in order to keep values consistent.