## Verification

The broad plan of development aims to provide the following requirements, ordered and categorised to clarify the main priorities of the project:

* **Impostor engine**
* A resource-intensive environment to test the impostor system
* A system to capture multiple objects at a distance into impostors
* A system to arrange the impostors in order and render them
* **Game experience**
  + WASD movement controls to enable environmental traversal
  + Balloon spawners to spawn balloons
* Balloons to shoot
* Weapon to shoot balloons with
* Game timer to provide a goal or the player and to continue to different configurations
* Game score count to give the appearance of the game and record data
* Cycling to the next random configuration when the game is finished
* **Experiment provisions**
* In-gameplay data collection of movement, balloon kill times,
* Starting questionnaire
* Intermission questionnaires
* In-game data recording (balloon lifetimes, player movement)
* In-game data export to .csv

The impostor engine is the core of the experiment as described in detail in the main paper.

The game experience is required to test the player response to the system. To ensure that other factors unique to the game (such as character movement or learning the controls) are of minimal influence to the results, a familiar first-person shooting setup is provided that resembles simple games of this genre.

The experiment features comprise the internal data recording system and external questionnaire UI. These features are essential to proving or disproving the hypotheses and identifying correlations worthy of discussion.

## Validation and testing

During development, a cyclic agile approach is adopted to ensure that the systems being implemented are working in their emergence. Due to the visual and kinaesthetic nature of the experiment, strategies such as unit testing are loosely employed wherein the developer occasionally compiles and runs the application to ensure the current phase is working correctly. In general, each user story is confirmed working when its stated effects appear to happen consistently. This process is described in detail on a per-phase basis.

During the **impostor engine** phase, the priorities focus on ensuring the system is generating the impostors correctly and at a high speed. This will be tested using a free-moving camera in an impostor-rendering system that the developer can ‘freeze’ in place. By freezing the impostors in place the developer is able to move around and ascertain that the correct layers exist and are appropriately replacing the 3D geometry. Similarly, Unity provides the ability to monitor the scene from any camera angle regardless of the in-game camera and this strategy will be used occasionally for the same purpose. The system will be confirmed correct when the radii of impostors layers are visibly behaving accurately, and changes in real-time when the developer changes the parameters.

During the **game experience phase**, the priorities focus on ensuring the shooting, score counters and timer work correctly. These will again be verified visually through iterative testing as part of the development cycle. Furthermore, the developer will make active attempts to escape the environment as the player character. The game will be confirmed ready when the above are fully functional when all four configurations can be played in sequence without crashes or issues in the above.

Finally, the **experiment provisions** will be tested in Excel. The data output by the game is checked by the developer to confirm that they match their experiences in the last session.

Following this development testing, the game will be considered ready to pilot-test. The pilot test will recruit three to four students of the BSC Computing for Games cohort and have them play the game. After this, the data file will be analysed. The usefulness of the data will be evaluated based on its variance. If the variance of results to a questionnaire’s likert scale between configurations is found to be below 2, the questionnaire will be re-evaluated and this may prompt a considerable revision. Specifically, it may be that the players notice the effects of the impostors, but do not recognise that the wording of the questions relate to them.

After pilot testing, the testers will be queried on their experience, noting explicitly the presence of the impostors, whether the player noticed the impostors, and how much of an effect they had. If the impostors are believed to have had a significant effect to the testers, yet it does not show in the questionnaire results, then the questionnaire may be revised to explicitly mention the impostors. Such a setup may introduce some unknown level of bias, but it is felt that that would be more useful than having a set of *responses* that simply does not recognise the impostors at all (even if such an *effect* is desired).

In fact, such a change is already under consideration. After all, as long as the player is not aware of the difference between each impostor configuration, it does not necessarily matter that they are aware so much as whether they notice the relative differences between the impostor and control configurations.