Conservatory Catch Project Description Two-Page Summary

Group 09: Jose Cuellar-Aguirre, Marcos Echevarria, Christian Garcia, and Nicholas Tryba

For our project, we are designing an Augmented Reality (AR) mobile game that will promote interest in the Garfield Park Conservatory (GPC). More specifically, we will be working alongside the Garfield Park Conservatory Alliance (GPCA), which is the nonprofit that works with the city to provide programming and resources for the conservatory and its visitors.

The requirements and acceptance tests are split up into sections, pertaining to various problems and concerns we expect to find down the line. These sections are: Functional, Data, Performance, Dependability, Maintainability and Supportability, Security, Usability and Humanity, Look and Feel, Operational and Environmental, Cultural and Political, and finally, Legal. We will separate this document by sections that pertain to them, giving a brief summary of what they're about and then diving into the requirements and acceptance tests of each one.

Functional requirements specifically deal with the functionality the system must have, and for our project, we have three requirements. The three deal with smooth scene transitioning, player reward tracking, and the user's location accurately being updated. The first one focuses on making sure there's some sort of transition between scenes as to not be jarring to the users, which the acceptance test addresses by making sure there's no less than 0.5 seconds of transitioning time between the scenes, as well as making sure there's an animation of sorts that transitions from the previous scene to the new scene. The second one deals with players being awarded their goods as they complete minigames, something that is essential, since it would drive down player count if they aren't able to get what they downloaded the app for. The acceptance test checks to make sure the users get their rewards, even if events were to happen, such as the game being closed, connection being lost, or even the servers going down and crashing. Last, but not least, we have the user's position relative to their GPS position, which is needed to function if we are to ever tell if they're actually in the Garfield Park Conservatory or not, and to be able to engage with the AR function that our product boasts. The acceptance test checks to see if the user's position relative to their GPS position is within a certain threshold, akin to "close enough" so that they're at or at least near the spot that the game thinks they are.

Data requirements deals with things that has anything to do with data, and we have two requirements revolving around data. One deals with data in the database, specifically plant names, and another deals with usernames. We don't want any duplicates for either, so we set up a system to prevent there from ever being any duplicates for either. If such a duplicate exists for plants, we throw an error and deal with the names that are duplicated.

Performance and Requirements deals with speed, accuracy, and capacity, and we have a requirement for each one of those criteria. For speed, we set an acceptance test that checks for whenever the latency is shown to be greater than 100ms, throwing an error and sending a bug report with the information on when it came to be, hopefully providing enough information to reduce the latency in that scene once the developers get it. For accuracy, we have an acceptance test that deals with the finger presses on the touchscreen of the users, checking to see if any scene transitions are wrong, or if any events are out-of-place. As for capacity, the acceptance test is dealing with multiple users active at once, seeing if the servers are able to handle 500 or more users at once, each navigating different scenes.

Dependability deals with reliability, availability, robustness, and safety-critical concerns. which we also have a requirement for each, although the first has two. I'll be combining the two acceptance tests from the first, as the first is similar to the last acceptance test, save for focusing on reducing the amount of crashes, and the second one focuses on preserving user reward upon a crash. For availability, we want users to be able to see their collection of plants even when they're not connected to the servers, which the acceptance test checks for. For robustness, we don't want the server to be down for too long, so we have our acceptance test check a function that would notify us whenever they are down so we can get them back up and running ASAP. Finally, for safety-critical concerns, we are concerned with the possibility of epileptic seizures, the acceptance test checking to see if there's any animations or transitions that are 3 seconds or longer, and if so, checking to see if any of them do not meet the safety standards of not possibly causing a seizure. If they do not meet it, we either scrap the transition / animation, or rework it to fit the safety standards.

Maintainability and Supportability has a lot more sections to it, all pointing towards making the project able to be maintained and supported throughout its lifetime. We have requirements that deal with server maintenance, broken game maintenance, ability to report bugs, being able to be played on mobile devices, being able to receive patches and updates, and even the idea of this product being applied to other plants/animals. With this, all the acceptance tests are focusing on making sure the developers are able maintain, support, and ultimately expand upon the game after its release.

Security deals with the data of the users, who can access them, and puts systems in place that prevents unauthorized users from accessing them. Tagged along with that, it also backs up the data that the game does collect in the case that someone is able to manage to break in and see the data, or in the off case that something in the main servers goes horribly wrong, they can rollback to the backed up data and minimize losses. Since this does deal with data though, one of the acceptance tests is making sure that users are fully aware what data is being collected.

Usability and Humanity provides many accessibility options to those who need them or to those who like them: things such as color blindness mode, subtitles and closed captions, changing font size, brightness, contrast, language, volume for specific in-game audio, voice-enabled commands, and many more things are provided in this section. Most, if not all of them can be found in the settings, while other things that provide help can be found in things such as the tutorial, or in the manual. The acceptance tests here just serve as a means to check to see if all of these features are fully functional. Look and Feel are more of an aesthetic thing, with things such as the color palette and the font / logos requiring approval from the client themselves. The only thing that doesn't require it is checking user engagement with the app. That particular acceptance test looks for how long users of the application stay at the GPC in proportion to how much they use the app while there.

Operational and Environmental has a few requirements, making sure that the application is designed for its particular purpose, as well as proving to be operational. The acceptance tests make sure the application is able to be downloaded and used by the majority of users from both the Google Play and Apple App stores, as well as the application's minigames and other area-specific functions are only functional whenever the user is at the GPC. It ties back to maintainability in the end, making sure the application stays operational for the future that is to come.

Finally, Cultural and Political as well as Legal requirements are similar in many ways, the bulk of it being something that can't directly be translated into tests without knowledge from other parties or entities. Here, the final set of acceptance tests do their best to make sure that the product adheres to many things, such as insurance industry standard, the Data Protection Act, and even internal political directives. The product must also make sure to not include any content that can prove to be offensive, derogatory, or culturally insensitive to any religious, ethnic, or social group. Once that is done, the product can finally be considered an asset to the client, which they can then take action to protect that asset under Intellectual Property Protection.