Administration Instructions

*UWA*

*Biosecurity Game*

By: Joel Dunstan

Personal Email: [jd427244@gmail.com](mailto:jd427244@gmail.com)

Work Email: [joel.dunstan@uwa.edu.au](mailto:joel.dunstan@uwa.edu.au)

biosecurity.are.uwa.edu.au

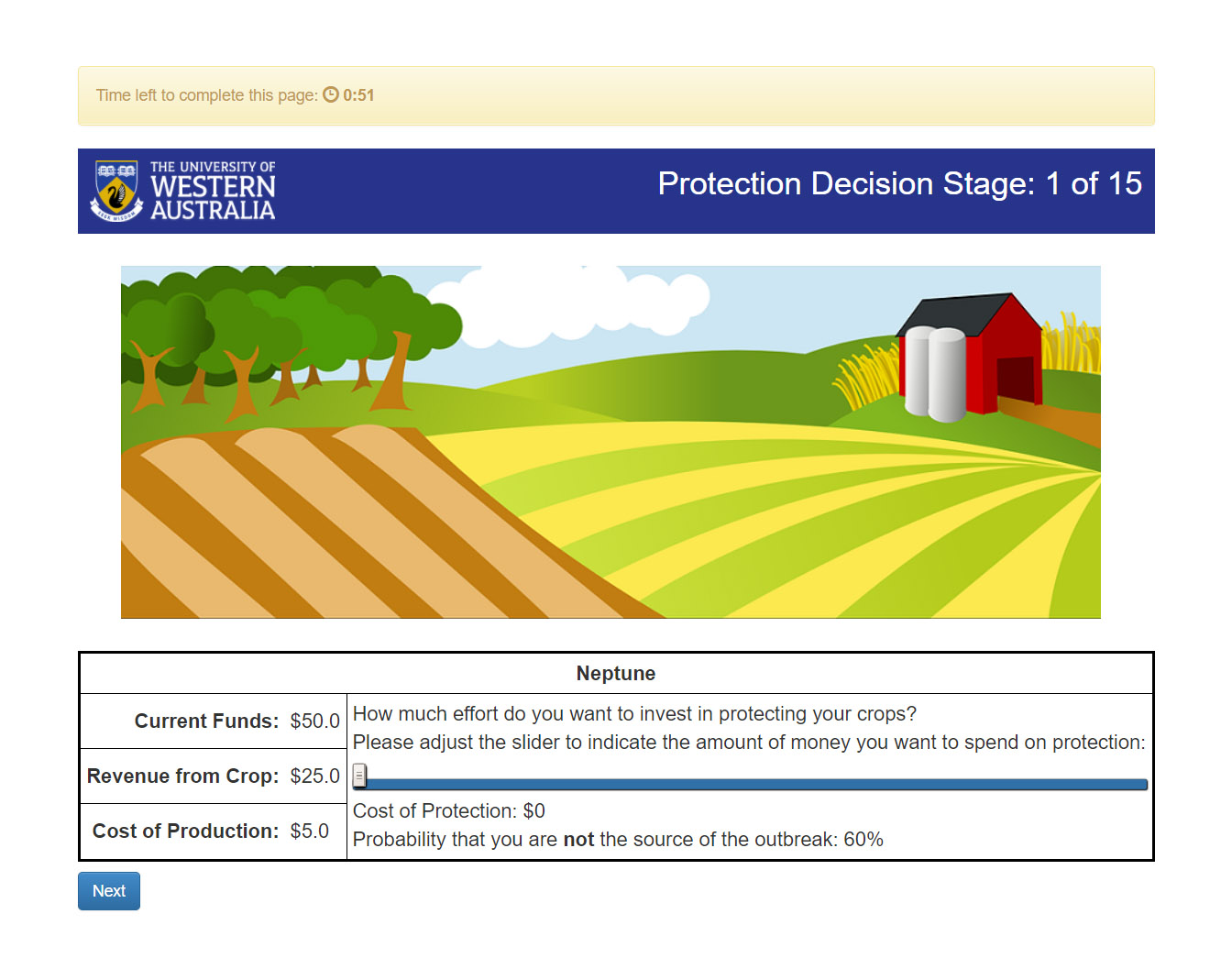


Table of Contents

[Overview 3](#_Toc507426028)

[How the Game is Played 3](#_Toc507426029)

[Session Configurations 11](#_Toc507426030)

[Freeform Communication Biosecurity Game 11](#_Toc507426031)

[Monitoring Biosecurity Game 12](#_Toc507426032)

[Pledging Biosecurity Game 12](#_Toc507426033)

[Monitoring + Pledging Biosecurity Game 13](#_Toc507426034)

[Approval on Contributions Biosecurity Game 13](#_Toc507426035)

[Being an Administrator 14](#_Toc507426036)

[Lottery Game 14](#_Toc507426037)

[Biosecurity Game 15](#_Toc507426038)

[How an Outbreak is Determined 16](#_Toc507426039)

[JavaScript Elements 17](#_Toc507426040)

[Questionnaire and Results 17](#_Toc507426041)

[How the Payoff is Calculated 18](#_Toc507426042)

[Using the Website 19](#_Toc507426043)

[The Demo Page 19](#_Toc507426044)

[The Sessions Page 20](#_Toc507426045)

[Configuring a Session 21](#_Toc507426046)

[Running a session 22](#_Toc507426047)

[Description 22](#_Toc507426048)

[Links 23](#_Toc507426049)

[Edit 24](#_Toc507426050)

[Monitor 25](#_Toc507426051)

[Data 25](#_Toc507426052)

[**Payments** 26](#_Toc507426053)

[Managing your Sessions: The Deletion and Archiving of Sessions 27](#_Toc507426054)

[Rooms 28](#_Toc507426055)

[Data Export 29](#_Toc507426056)

[Server Check 30](#_Toc507426057)

[Being a Developer 31](#_Toc507426058)

[Python and oTree 31](#_Toc507426059)

[Setting up a Server 32](#_Toc507426060)

[oTree Tags 32](#_Toc507426061)

[Linux 33](#_Toc507426062)

[Windows 35](#_Toc507426063)

[Resources 36](#_Toc507426064)

[Acknowledgements 36](#_Toc507426065)

[Glossary 36](#_Toc507426066)

[References 38](#_Toc507426067)

# Overview

This document will cover the use, modification and maintenance of the website, biosecurity.are.uwa.edu.au. This website is run from a UWA virtual server running with Dual Cores with Hyperthreading for a total of 4 threads/virtual cores and 16GB RAM. The instructions will first show you how to create a game, and along the way explain what each page is for. This will be done with an example game as shown from the website in its current state during May 2017 onwards. Secondly, the instructions will give you some idea on how to modify the game if needed, this will include links on places to learn python, oTree, Django etc. Finally, the instructions will explain how to maintain the website.

I heavily recommend you read the oTree documentation before reading this document, the oTree documentation is located here:

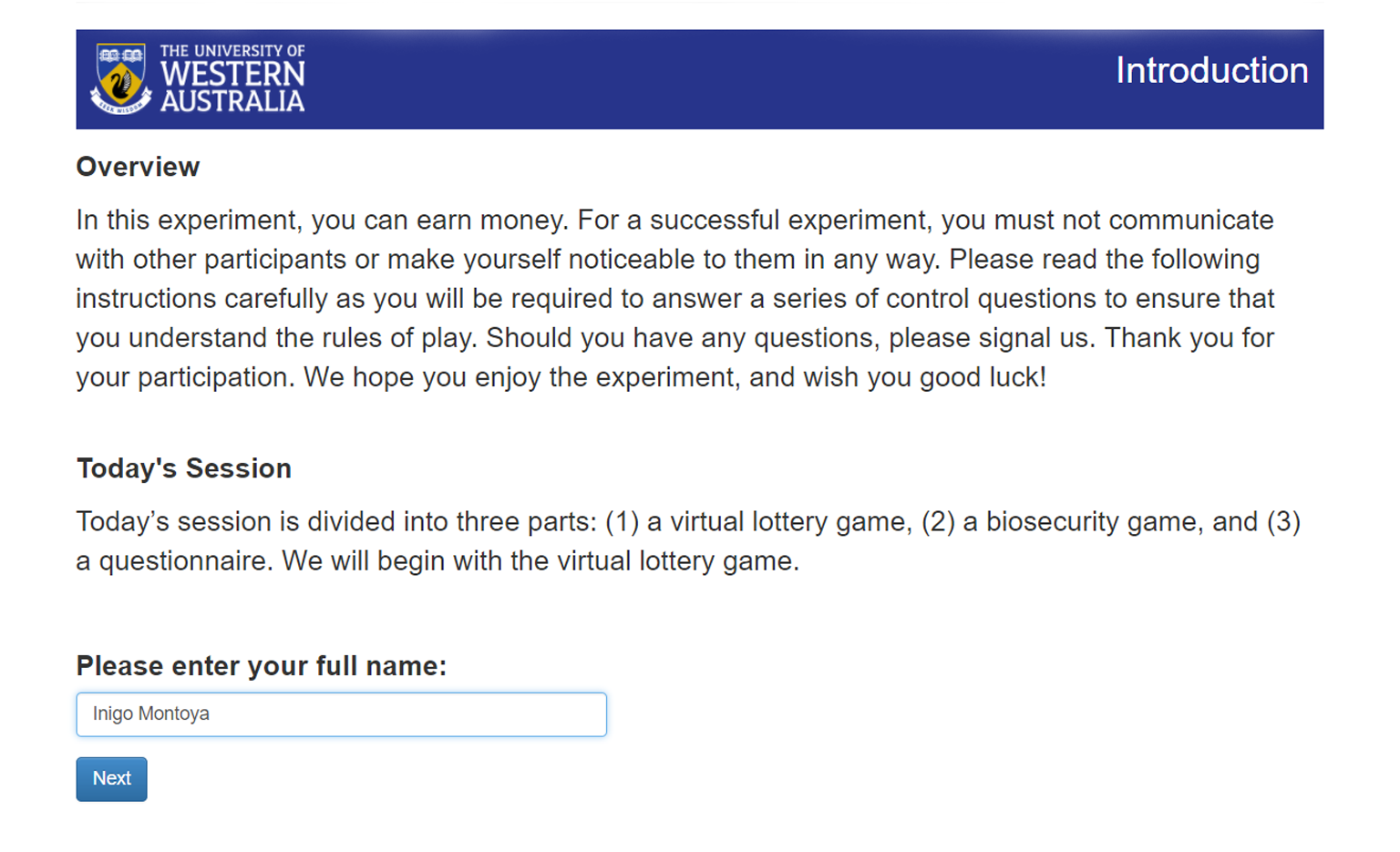
[oTree Documentation](http://otree.readthedocs.io/en/latest/index.html)

The documentation, includes tutorials, the description of oTree’s models, concepts and setup.

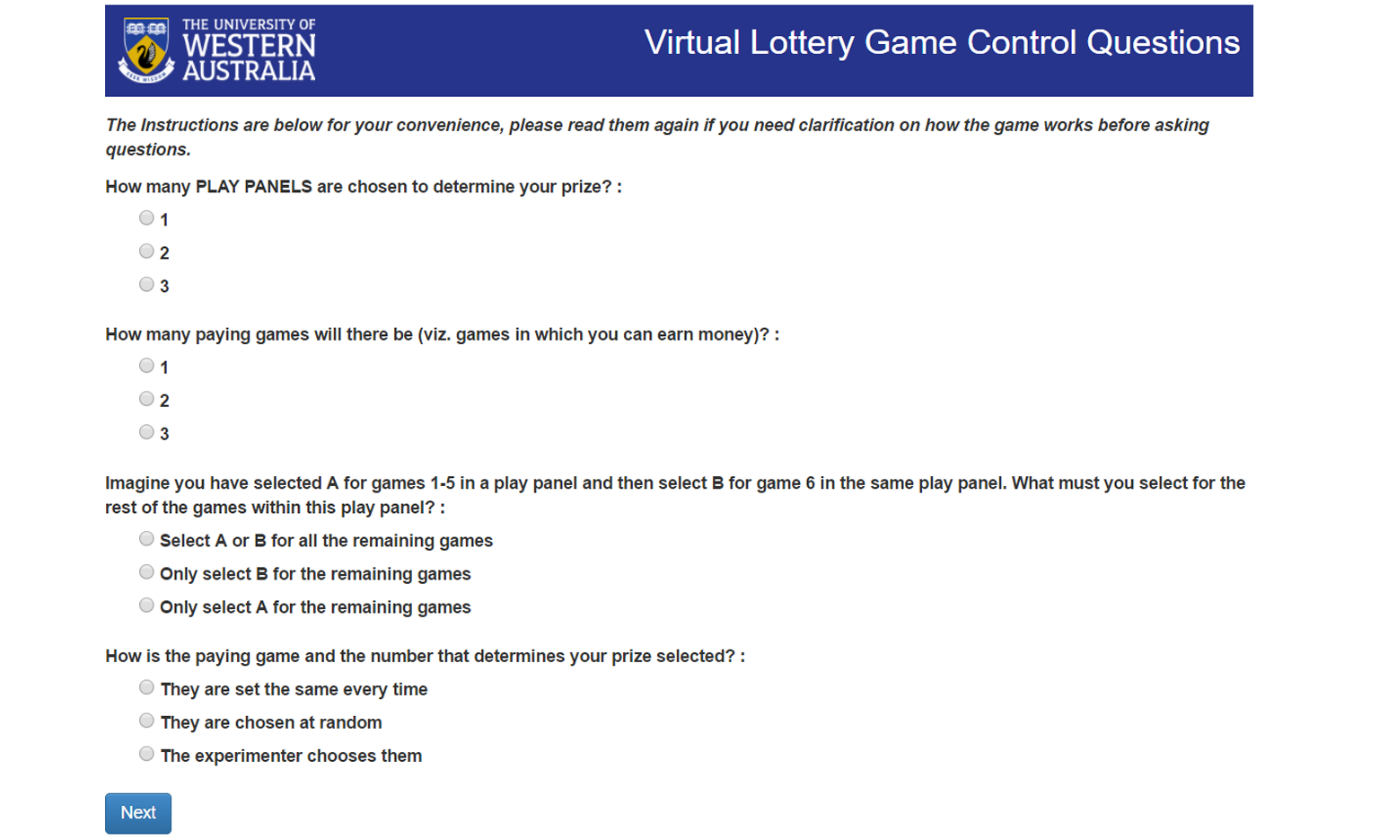
To learn how to use the website and administer a game, you must first learn how the game works and the way the game is played.

# How the Game is Played

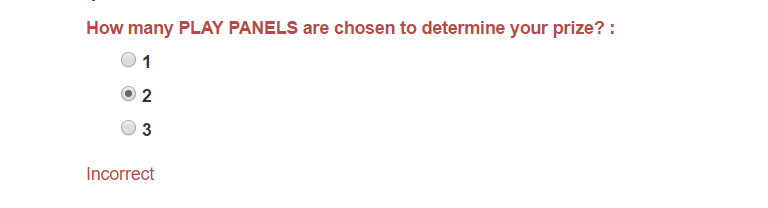
To explain how the game works and how to play the game we will look at the game from the perspective of a participant in a game you’re administrating. We will call this participant, **Inigo**. Inigo has just sat down in front of a computer you have set up in such a way that you have opened a link on Google Chrome (as per the recommended web browser in the Test Document). Inigo will be greeted with this screen:

What’s happening here while Inigo is reading the introduction to the game is that the UWA virtual server (which we will now call the Web Server) has received a http request from the computer Inigo is on. The web server receives this and in turns runs any Python code associated with this page in the models and views files and the Django, HTML and CSS code in the template that generates this page as you see it on a web browser (PCMag, 2017). This is how all the page’s work, some like the round page in the biosecurity game are more complicated and require JavaScript to change elements dynamically on-screen, the web server handles all the calculations, the only thing a participant like Inigo needs is a web browser and a device that can run a web browser.

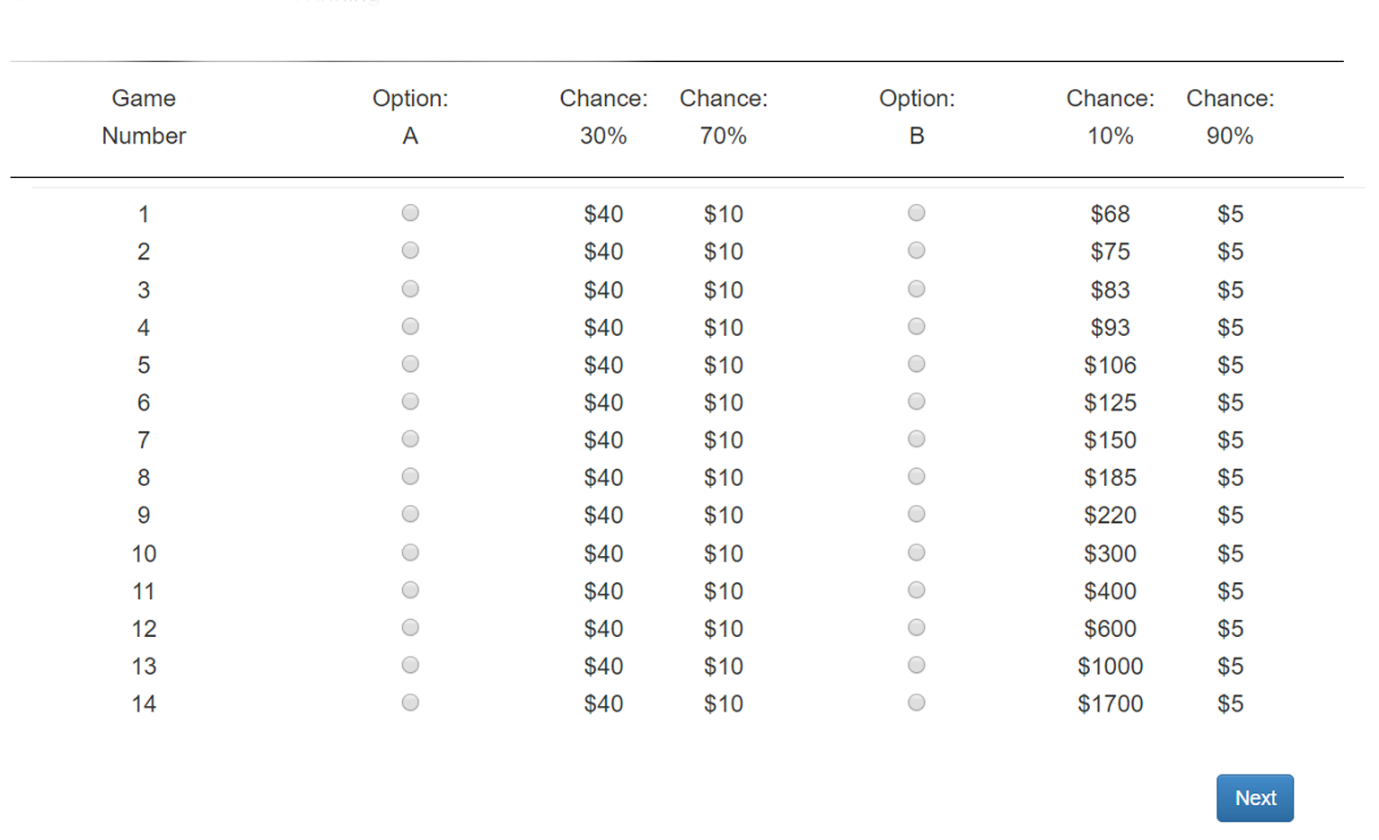
Now back to Inigo, he has hit the ‘Next’ button by now which advances him to the next page, he will read the lottery game instructions (please read them [here](https://1drv.ms/w/s!AphuEhKWv8YUg4QgL2TgiEDscFOtcQ) or by playing the game). After Inigo has read the instructions, he will wait for all other participants to finish reading the game on a waiting page. After everyone has reached the waiting page, the game will load a control questionnaire that test’s his understanding of the lottery game. This will appear as so:



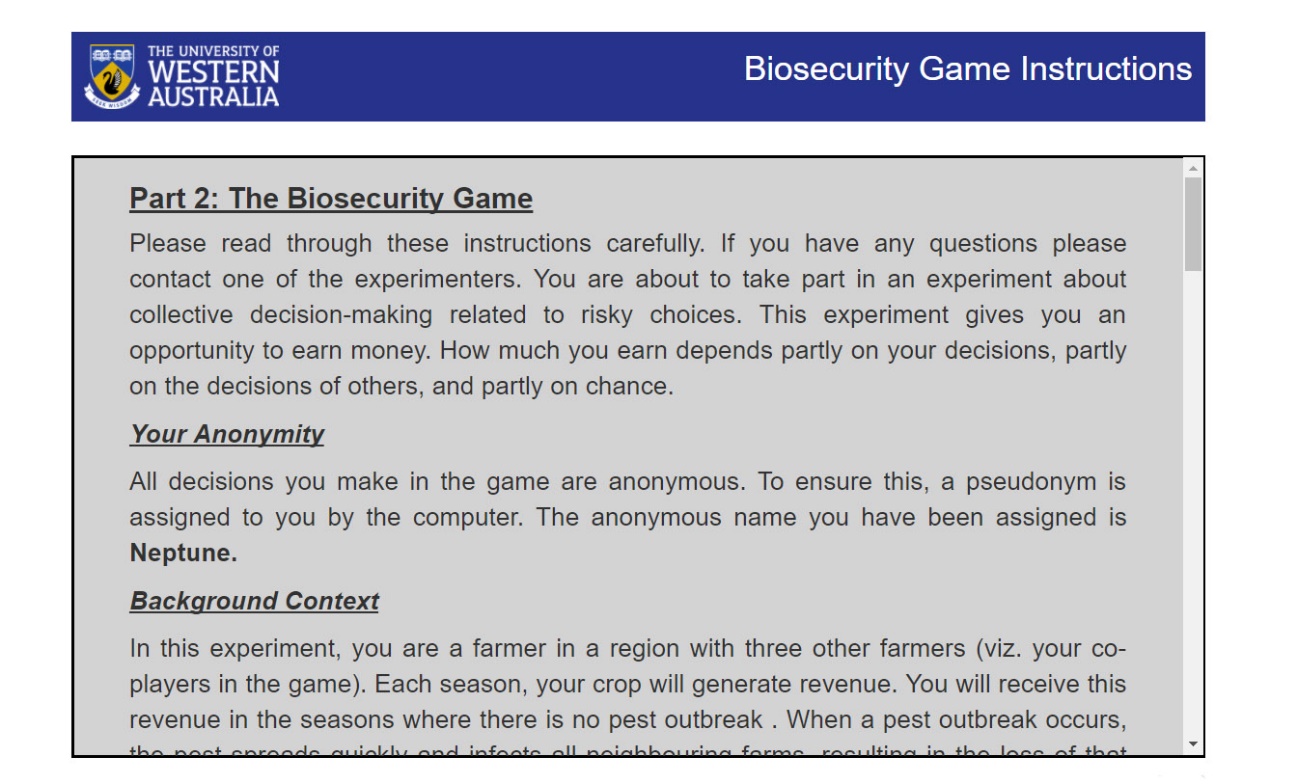
On this page, Inigo will choose answers and then hit next, if any of them are incorrect, the game will tell him after he clicks next. The game specifies which ones are incorrect by changing the text for the questions to red and labels them as incorrect underneath the question, e.g.



Inigo will only advance to the next page when he answers every question correctly, the instructions are provided below so Inigo can refer to them if he needs to remind himself how the lottery game works. Once everyone has correctly answered the questionnaire, Inigo and his group members will advance to the lottery game play panels where Inigo will choose his options for the lottery game that could determine his prize (or loss) for the lottery game. If you missed the link I provided you for the lottery game instructions, here it is again, [Lottery Instructions](https://1drv.ms/w/s!AphuEhKWv8YUg4QgL2TgiEDscFOtcQ), you’ll need to read them to understand how the game works.

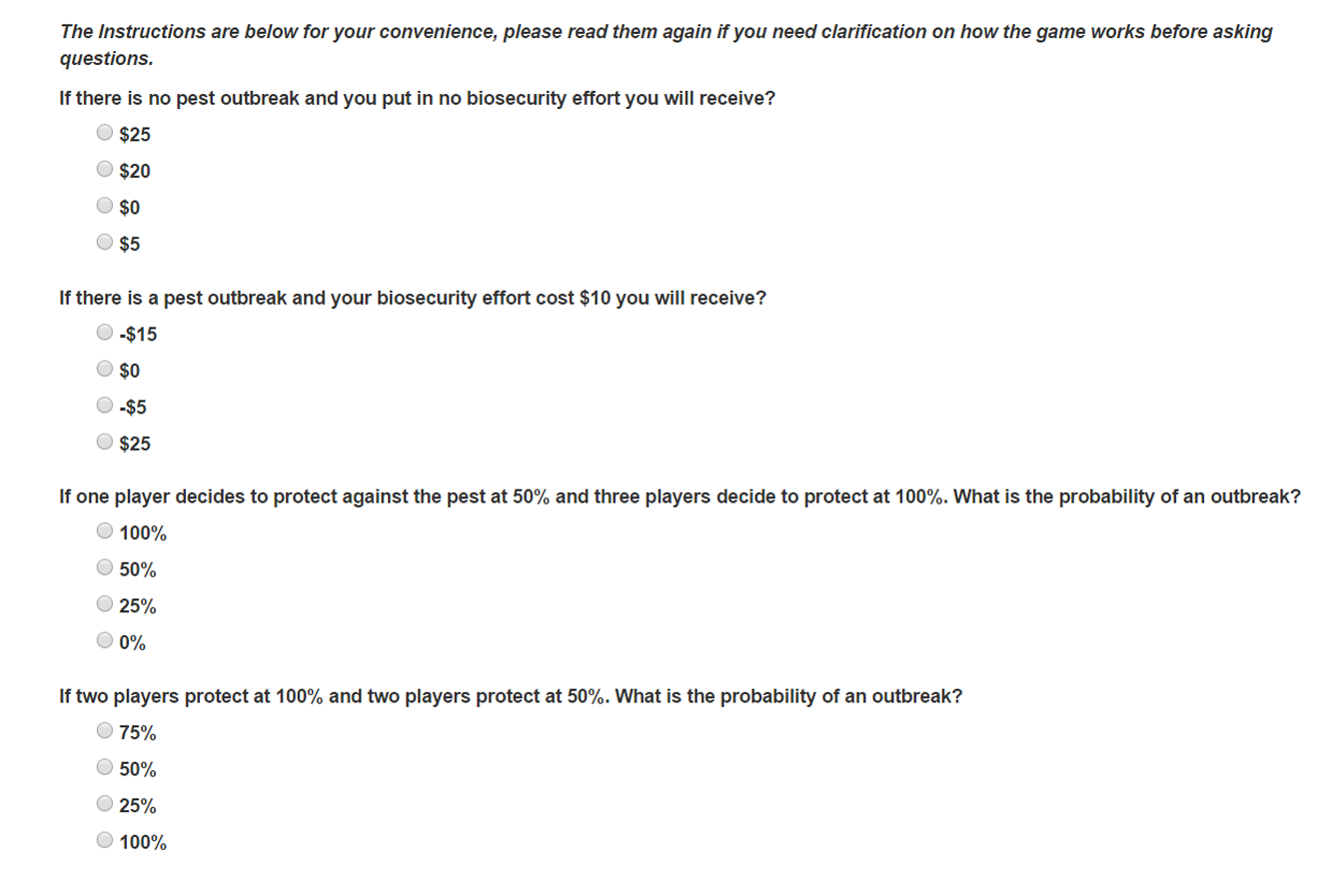
The play panels will appear as so:

Inigo will choose his options, to assist Inigo there is JavaScript used in the template to ensure that once he chooses option B, the page will automatically select option B for the rest of the games. Since the lottery game can be played individually, Inigo will complete all three play panels without waiting for other participants to finish each play panel. A waiting page will only appear once Inigo has completed all three play panels and has therefore finished the lottery game.

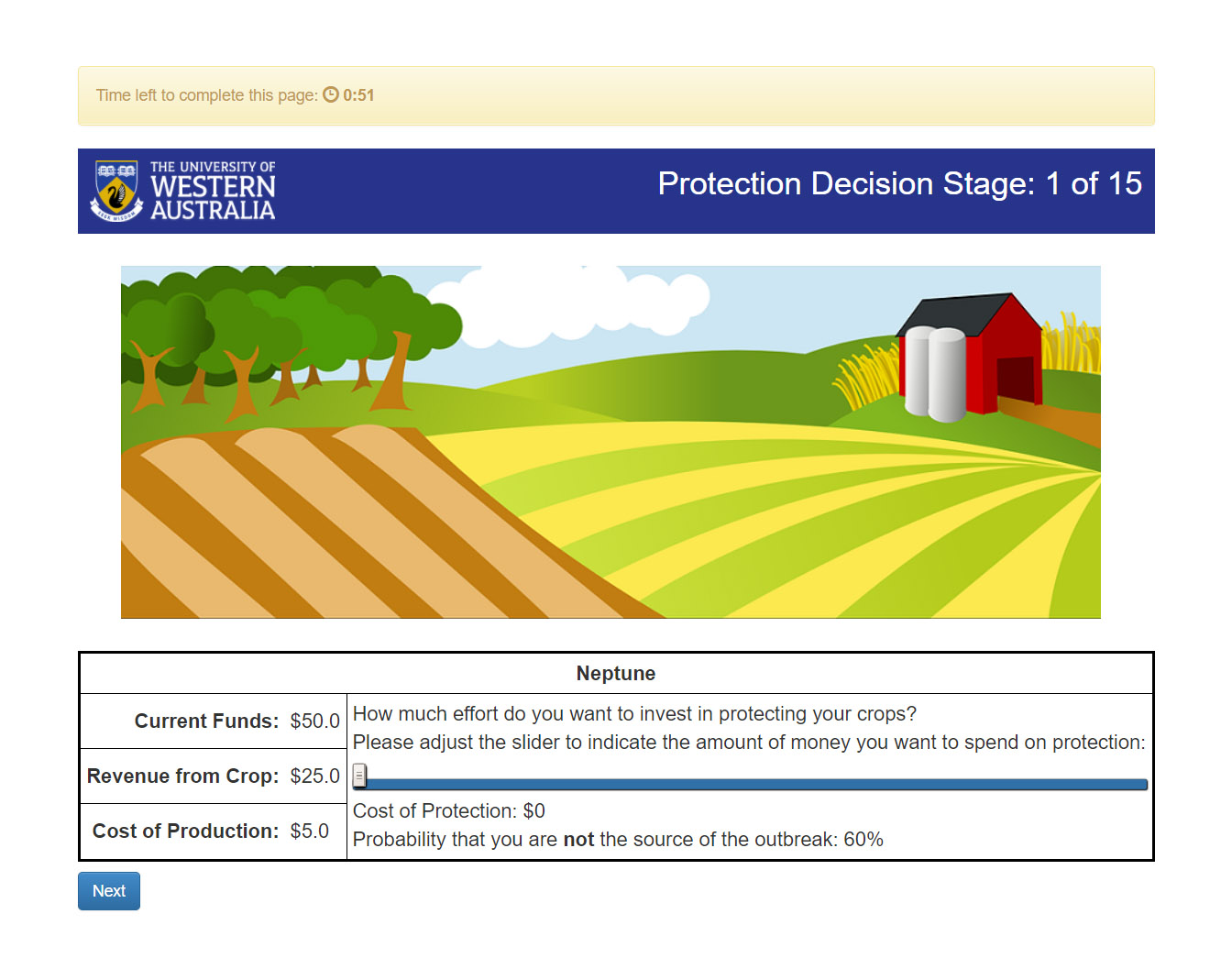
Once everybody has finished the lottery game, the game will present the biosecurity game instructions to Inigo. The instructions will assign an anonymous name to Inigo, the name is chosen randomly from 25 names of planets and moons, Inigo received the anonymous name ‘Neptune’.

The instructions Inigo sees will be dependent on what session configuration of the game you set up for him. The different session configurations unlock different modes of the biosecurity game, this in turn requires different instructions which will be added into ‘The Stages’ section of the game. The session configuration Inigo is currently in, is the ‘Basic Biosecurity Game’, which is the barebones version of the game and will occur as you will see in this section. For the description of other session configurations, please refer to [here](#_Session_Configurations). I encourage to read the instructions for the base version of the game yourself, get them [here](https://1drv.ms/w/s!AphuEhKWv8YUg4sDGR_IPMsebia6Sg).

After Inigo has read the instructions, the game will present him with a control questionnaire to ensure his understanding of the biosecurity game. These control questions will have same appearance as the control questions for the lottery game, like the lottery game, they will also tell Inigo if he got any incorrect. The control questionnaire does include questions on how joint probability works to ensure participants know how an outbreak is determined. Luckily, Inigo is good with maths and can handle joint probability, however just be aware some of your participants may not be so lucky.

These questions are retrieved from a CSV file inside the main project folder, an administrator can change these and their options, however, doing so means that one should reset the database used to hold all the data and restart the website. So before editing the questions, it is imperative to check that you have all the data for the games inside the database currently, once the data is gone, **there is no going back, there will be no way to retrieve them.**

After Inigo and his fellow group members have completed the control questionnaire, the biosecurity game begins. In this session configuration, there are no stages before Inigo will make his contribution towards the biosecurity effort, thus Inigo will be greeted with page below, please take note of the labels, they explain the elements of the page.



The slider is what a participant will use to determine the amount of effort they will contribute to the biosecurity effort. When the slider changes, the cost of protection and the probability that someone is not the source of the outbreak also changes using JavaScript to dynamically change these elements.

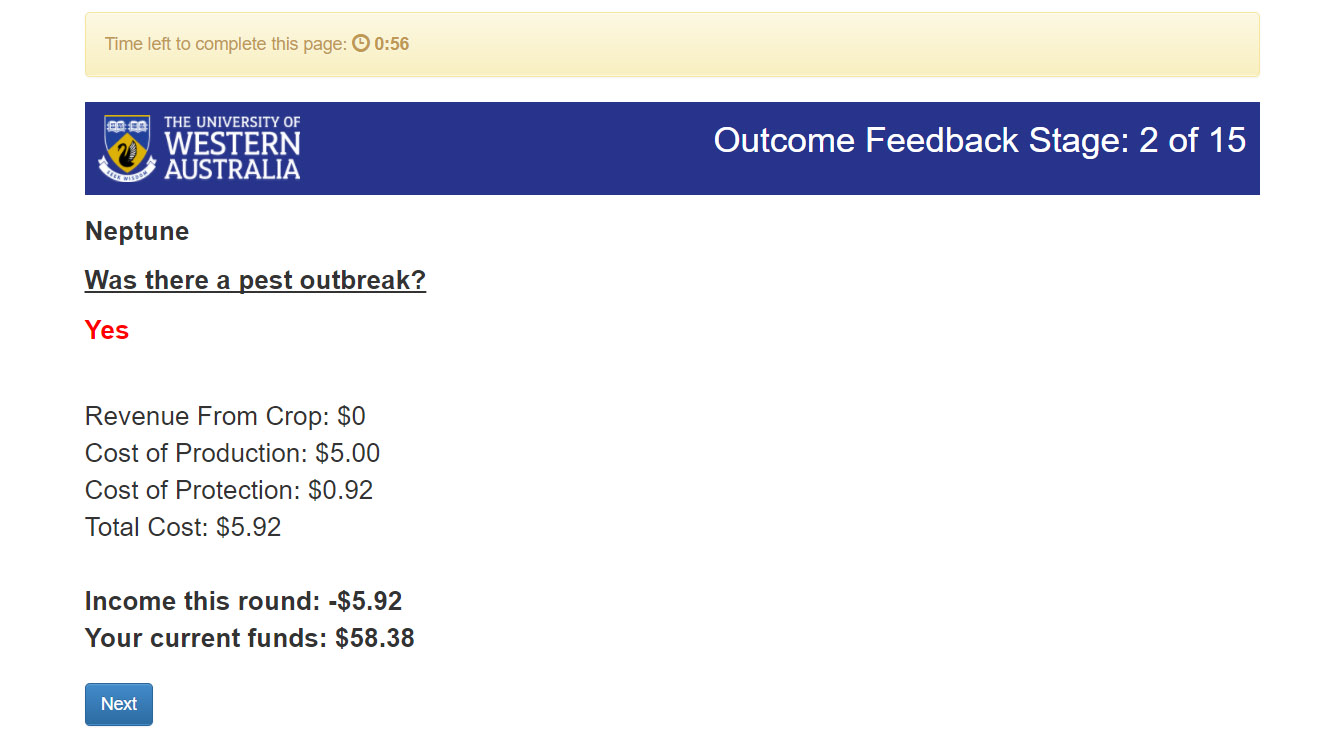
The participant’s anonymous name, while it has no significance in this session configuration, ‘Basic Biosecurity Game’, the anonymous name will be used in every other session configuration.

A timeout is put on the page, to ensure the game continues even if someone decides not to act. The timeout on the round is 1 minute and 30 seconds.

This is the round number, participants can keep track how many rounds there are left in the game.

Inigo and his fellow group members will use the slider to determine how much they will contribute towards the biosecurity effort in terms of dollars. The probability of Inigo not being the source of the outbreak is also shown, Inigo with his math skills could work out the groups joint probability of an outbreak by taking his own probability that he is not the source of the outbreak putting this to the power of 4, and then subtracting the result from 1 (e.g. 1 – 0.64 = estimated group’s probability of an outbreak). However, Inigo should not take too long to make his decision, there is a timeout of 1 minute and 30 seconds. Should Inigo fail to hit the next button thereby deciding on his biosecurity effort, the game will randomly choose the biosecurity effort (a value between $0 and the Maximum amount of protection). Throughout testing it was better for participants to make decisions rather than let chance completely decide their actions, not to mention it renders the experiment meaningless if participants start randomly choosing their actions.

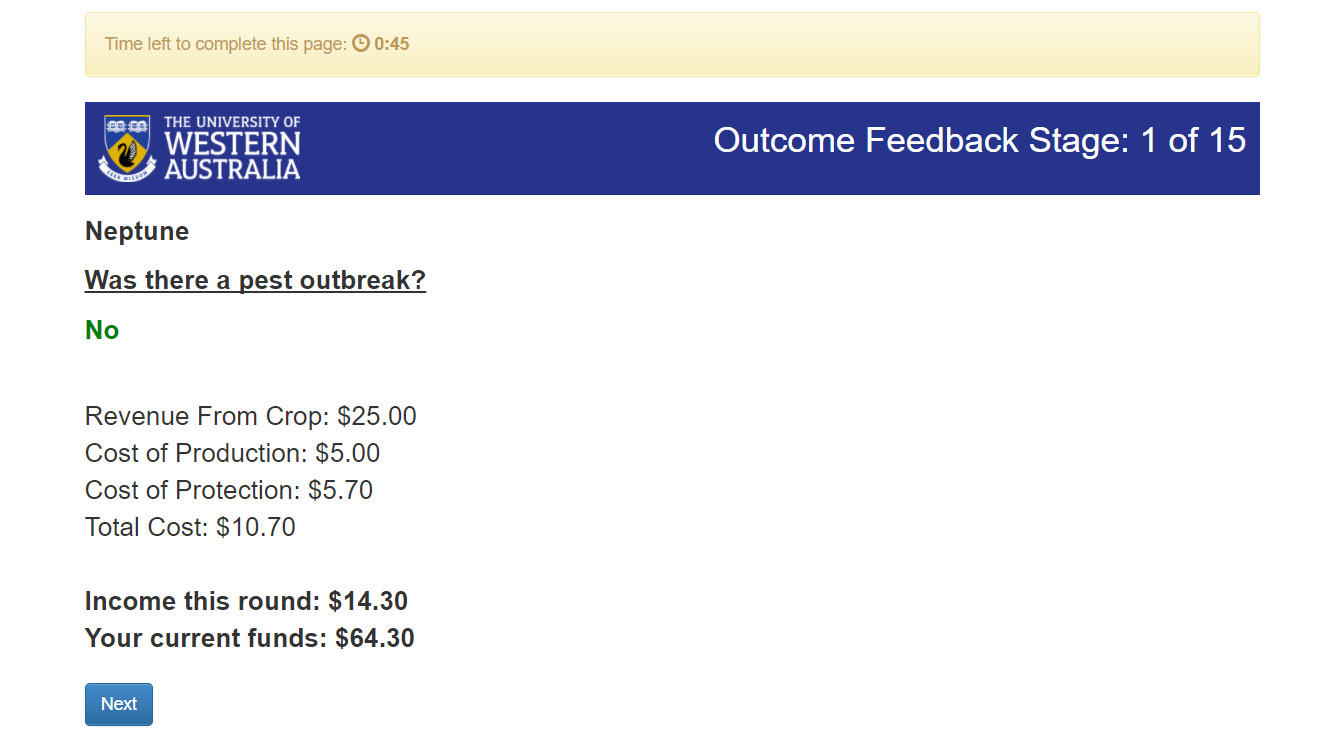
If Inigo decided his biosecurity effort before the rest of his group members he will wait until the rest of his group members have made their biosecurity efforts. After everyone has made their biosecurity contributions, Inigo will be presented with the results page. The results page will show the income from the round, and whether there was an outbreak. If there was an outbreak Inigo will see the results page as below.



The anonymous name appears on every results page, this will be of big significance on every other session configuration where participants will see everyone’s biosecurity efforts including their own. In other configuration participants can use this anonymous name to communicate or give approval in some way. Thus, it is important, just not in this configuration.

Also, there some changes in the results page used to help participants immediately notice whether there was an outbreak or not. Take notice of how the yes is in red, and the no is green. Also, income is highlighted in bold so it’s one of the first things participants will look at, immediately indicating a loss or profit to the participant.

Inversely, if there wasn’t an outbreak, then Inigo will see the results page as below.

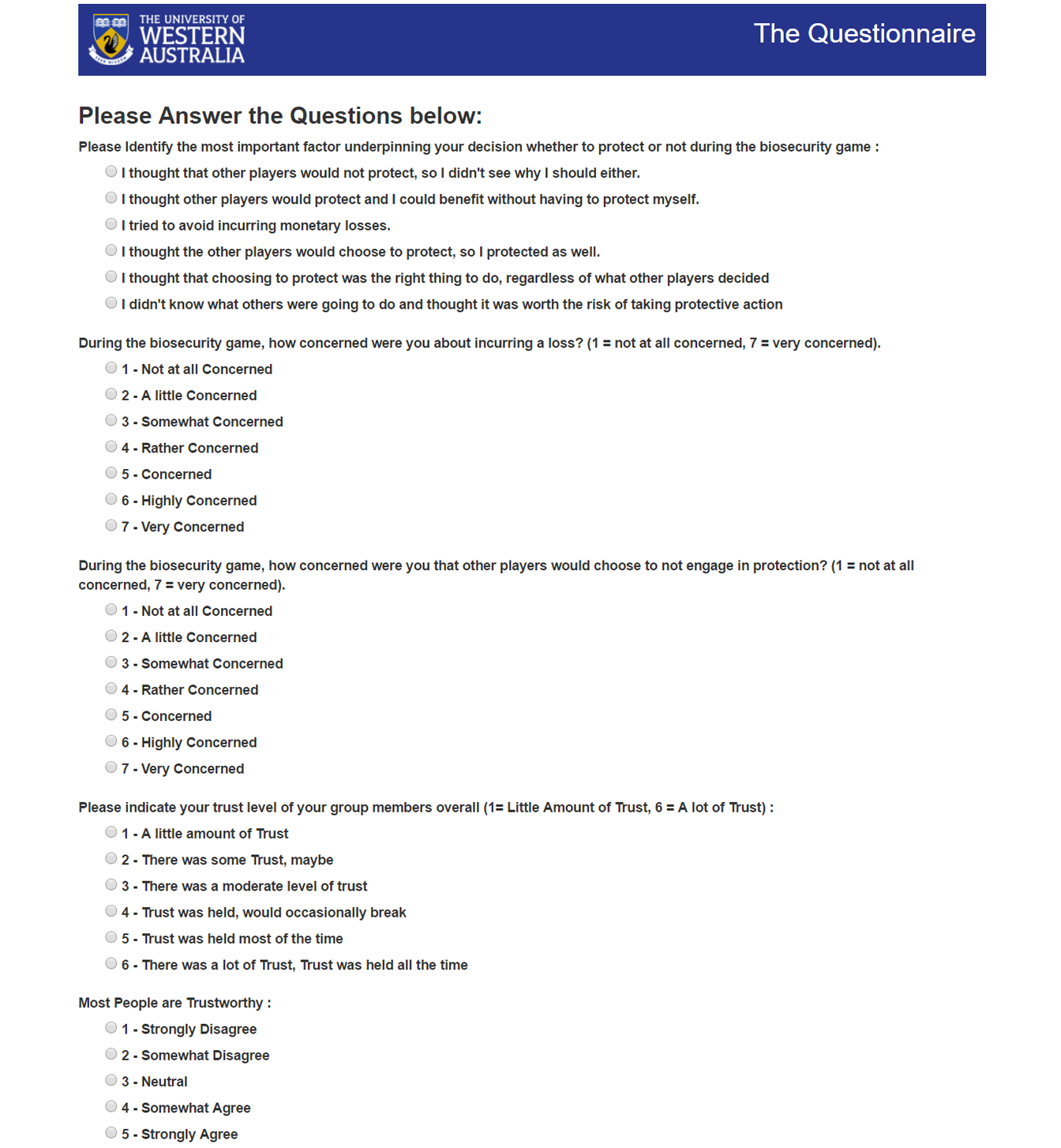


The total cost is indicated, should participants feel the need to see all the information without having to work it out.

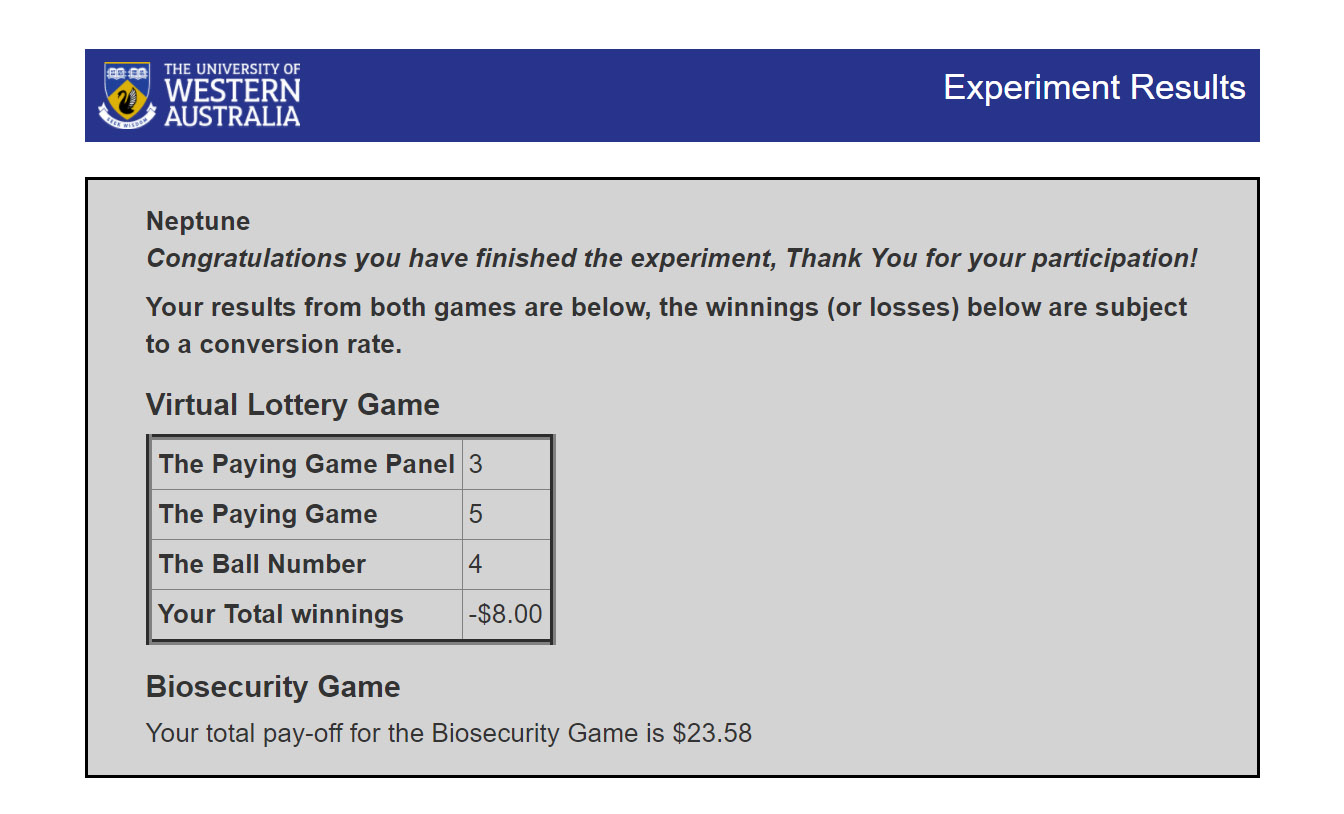
As the game will continue into round 5 and above you will notice participants will only take note of all the words in bold, and the yes or no, and the game will progress quickly from here.

Please take note of the differences between the above images.

Inigo, after seeing his first results page will continue the game for another 14 rounds, each round Inigo will decide his biosecurity effort alongside his group members and see the results. Finally, Inigo will finish the biosecurity game after completing 15 rounds with either a profit or loss.

Inigo will now proceed to a questionnaire which will ask him questions about the biosecurity game, his thoughts behind his decisions and how much he might trust his fellow group members etc.; questions about his gender, age, suburb and income are also asked.

Once Inigo and his group members have completed the questionnaire, they will have completed the game in its entirety and will be presented with the results of the lottery and biosecurity game.



Inigo and his group members will now receive their winnings from the game as per a conversion rate set by yourself in the currency of your choice.

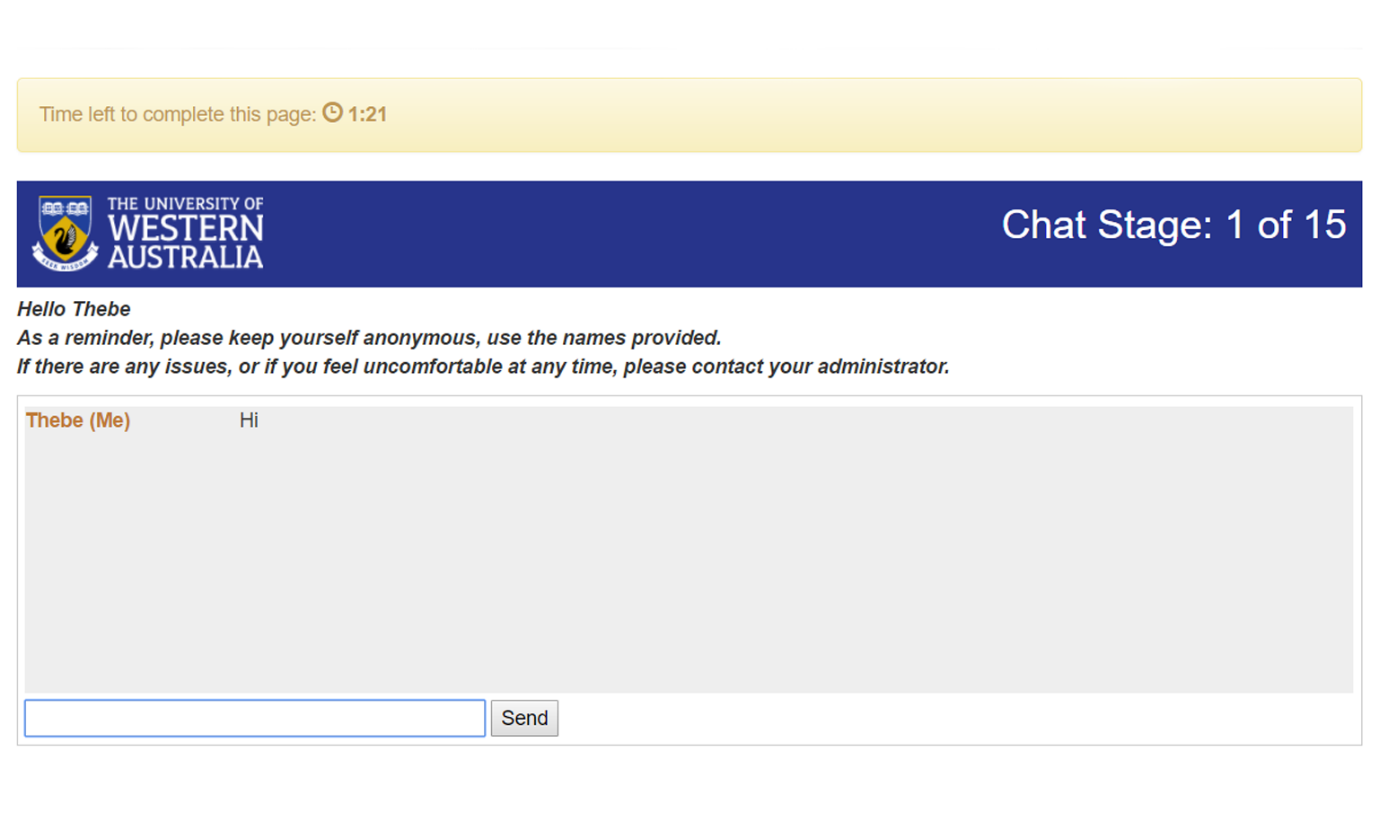
# Session Configurations

As I mentioned in the previous section, there are multiple session configurations for the game. The difference between the session configurations can only be seen within the biosecurity game, the lottery game and the end questionnaire always remain the same despite a change in session configuration.

Some of these session configurations will make the game easier to play and allow participants to communicate and coordinate with each other to increase everyone’s profit. However, they could also potentially cause larger rifts between participants in contrast to the base version, thereby increasing the loss instead. How the configuration affects the profits or losses of participants could be interesting in these cases. These session configurations are based on how farmers might coordinate their efforts and communicate with each other when protecting the community and themselves against biosecurity threats.

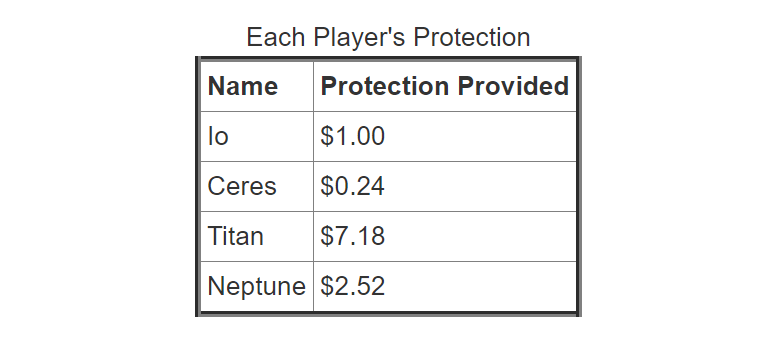
## Freeform Communication Biosecurity Game

This session configuration is exactly like the barebones version of the game except for an added chat stage which allows participants to anonymously communicate with each other using a chat box. The chat box will appear on the 1st, 6th and 11th round (every 5 rounds). No filters are present in the chat box, you can keep an eye on their conversation by clicking one of the links of the participants and watching it in real time. Hopefully, participants will remain civil and keep the chat relevant to the game, but there is no guarantee.



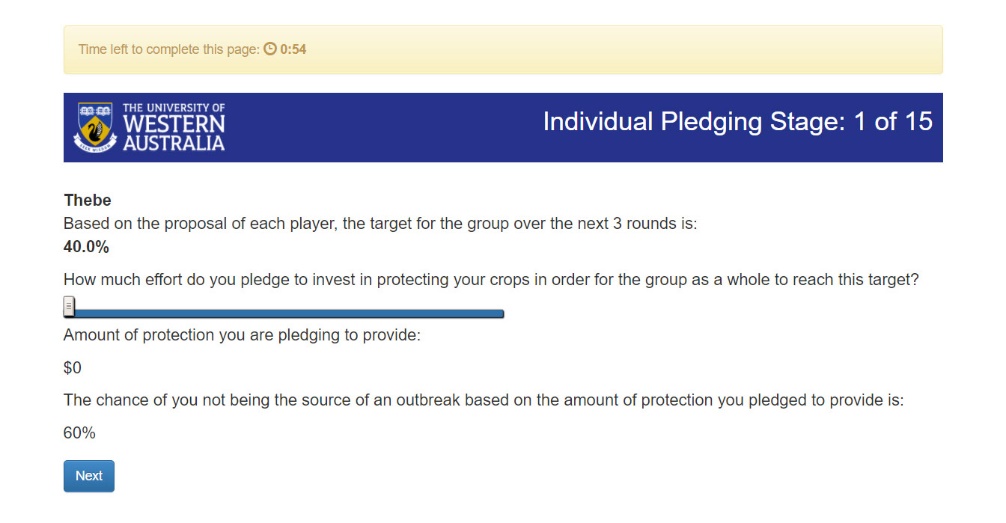
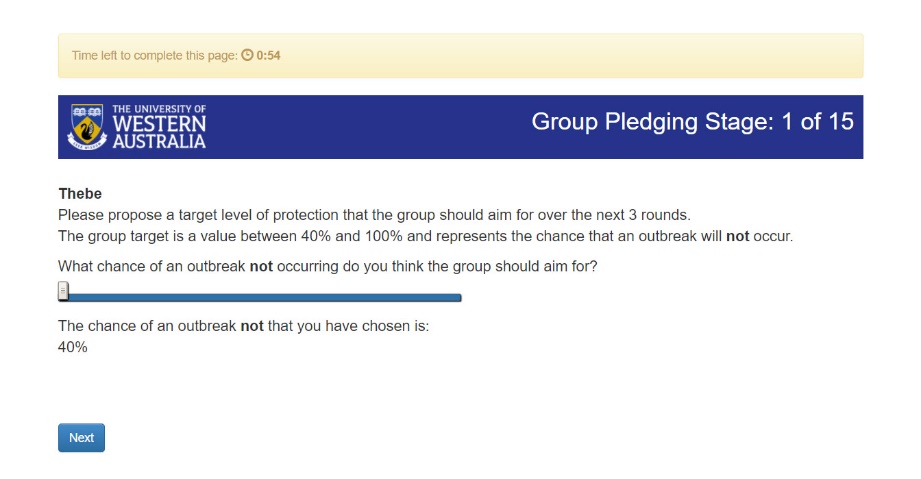
## Monitoring Biosecurity Game

This session configuration is like barebones version of the game except participants can see everyone else’s biosecurity efforts including their own during the results page of each round. However, participants have no way of showing their approval of these efforts.



## Pledging Biosecurity Game

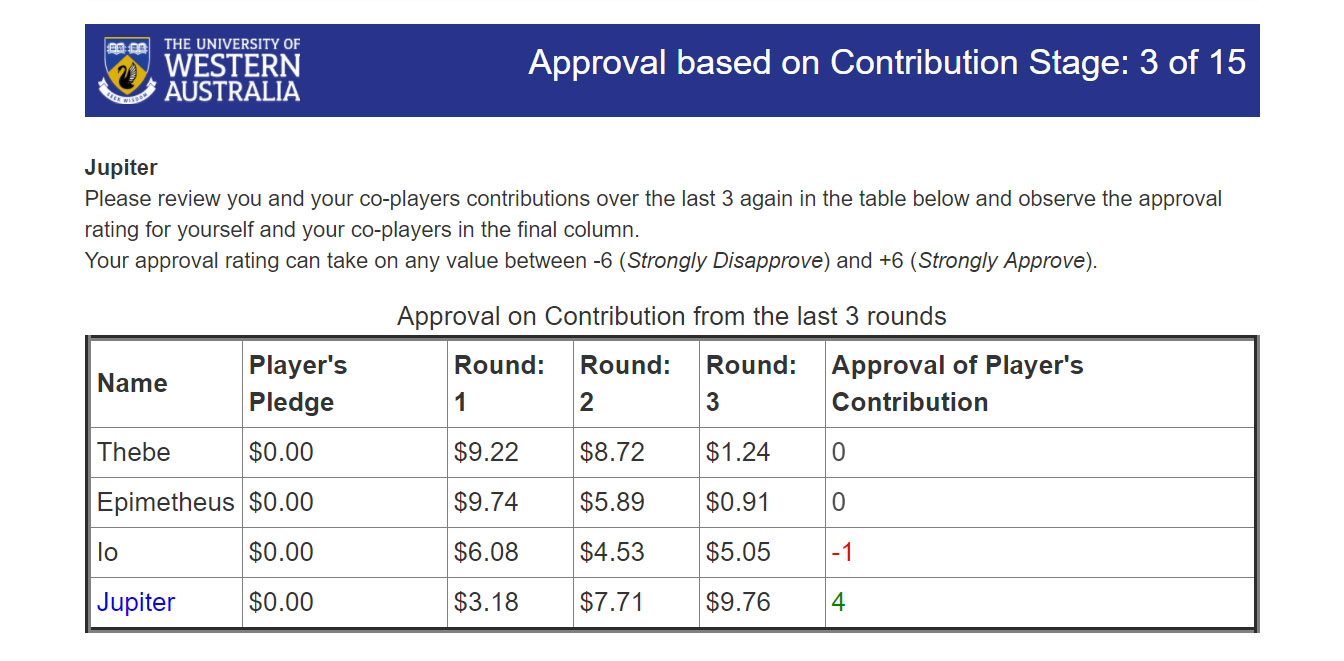
This session configuration adds two stages to the barebones version of the game, a group pledging and an individual pledging stage. The group pledging stage happens at the start of every third round including the first round. In this stage participants choose what they think the joint probability of the group not having an outbreak should be.

Once participants have chosen their ideal probability of the group not having an outbreak, it will take the median of every participants ideal group target which produces the official group target that everyone should strive for. The official group target will appear in the individual pledging stage which occurs straight after the group pledging stage. In this individual pledging stage, each participant will pledge to do a certain amount of biosecurity effort per round for the next 3 rounds to reach the group target. However, there is no way for participants to show approval of everyone’s pledges during the round where they decide their biosecurity efforts and see everyone’s pledges including their own.

## Monitoring + Pledging Biosecurity Game

This session combines the Monitoring and Pledging Biosecurity Game session configurations together to have a game where participants can make pledges but also track whether other participants (including themselves) are keeping to their pledges due to participants being able to see everyone’s biosecurity efforts after each round.

## Approval on Contributions Biosecurity Game

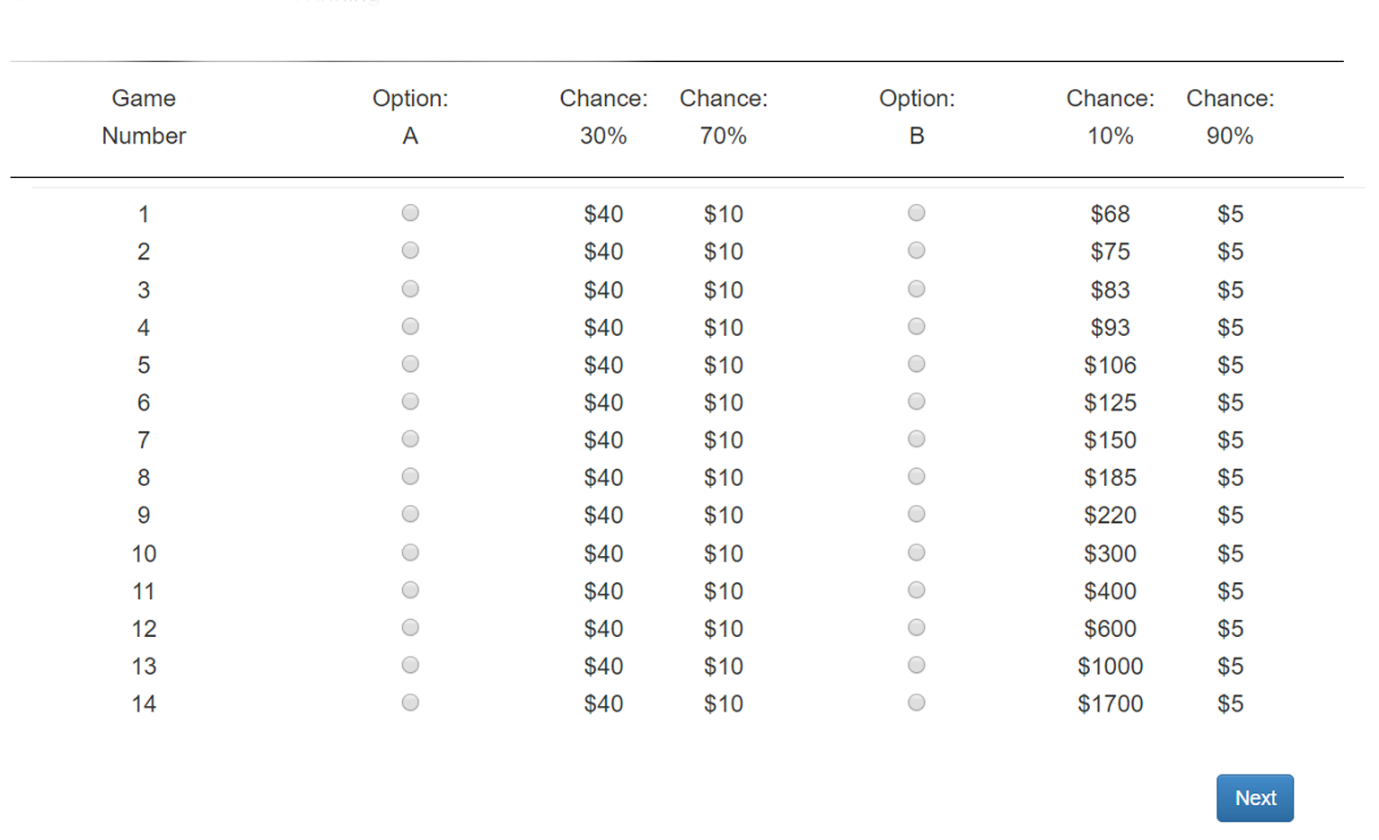
This session configuration, like the Approval on Pledges Biosecurity Game, also adds an approval element to the Monitoring + Pledging Biosecurity Game. The approval focuses on the contributions of participants. In this session configuration, every third round will contain an Approval on Contribution Stage. During this stage participants will use the table to indicate their approval of each participant including themselves on a scale from -6 (Strongly Disapprove) to 6 (Strongly Approve). After deciding the approvals for each participant, the game will take the group approval (an average of all the approval ratings) of each participant and show them these values on the next page to ensure participants see these approvals before making anymore actions such as a pledge or biosecurity effort. The group approval ratings of each participant will be seen during each round.

# Being an Administrator

This section will focus on how Inigo’s game would look from an administrator’s perspective, or *your* perspective. As such this section will cover the specifics of how the game makes it decisions (random or otherwise), how to administer a game in terms of using the website biosecurity.are.uwa.edu.au and how to modify how the game works (or appears) without changing any of the underlying Python that the game relies on for its functions, equations and logic.

I will start with the specifics of the Lottery Game, Biosecurity Game and the Results containing the Questionnaire, and how participants are paid their reward. The sections below will explain the underlying functions and equations for each game, including how each participant’s reward is calculated based on their performance in the games.

## Lottery Game

In the Lottery Game, each participant receives a total of 3 Play Panels, these play panels have up to 15 games, there is a total of 35 games and each game has two options, A or B.

Referring to the play panel above, I will explain how the lottery game chooses the winning game and how each participant determines how much they win. First, the lottery game randomly selects 1,2 or 3, each number representing a play panel. In this case, it randomly selected 1, thus it chose play panel 1. Secondly the game will randomly generate a number between the lowest game number to the highest game number inclusive, in this case it’s a random number between 1 and 14 inclusive. As an administrator, you should be aware that on play panel 1, it will only select games 1 to 7 due to the high monetary values included from games 8 to 14. If you’re wondering why this is the case, consider this, even when a conversion rate of 0.1 is applied, you could be paying participants the equivalent of $170 in the currency of your choice when a participant wins through option B on game 14.

Now back to the game, the random game that was chosen is game 2. Now, in this lottery game, a number between 1 and 10 is randomly generated, it is this number and the option chosen by a participant that determines exactly how much a participant wins. For example, in game 2, let’s examine two scenarios, in the first scenario, a participant chooses option A. For a participant to win $40, the randomly generated number must be between 1 and 3 inclusive, out of a possible 10 numbers, hence a participant has a probabilityor 30% chance to win $40. If the randomly generated number was between 4 and 10, then the participant would win $10, hence a participant has a probability or 70% chance to win $10.

The second scenario for game 2 is a participant choosing option B. The scenario will play out much like the first, however, for a participant to win $75, the randomly generated number must be 1, thus a probability or 10% chance of winning $75. Thus, leaving a probability or 90% chance of participant winning $5.

The probabilities of the random number between 1 and 10 apply throughout the entire play panel. For example, for every game inside play panel, option A always has a 30% chance of winning $40 and option B will always have a 10% chance of winning the higher monetary value.  
  
To summarise…

A random number is generated between 1 and 3 (inclusive) this is the **winning play panel**.

A random number is generated between the lowest and highest game number, this is the **winning game.**

A random number is generated between 1 and 10 (inclusive), and a participant will choose an option in the winning game. It is these two variables that determine a **participant’s prize**.

To ensure participants play the lottery game correctly, there are safe guards. The first being that when a participant chooses option B, it will select option B for the rest of the play panel, this will be explained in the instructions for the game. Secondly, the game numbers, monetary values and text is read from a CSV file, making it harder for someone to edit the values using a developer mode inside a web browser to change the values thus making the data invalid. Finally, there is a lottery game control questionnaire to ensure understanding of the game.

If the lottery game runs as expected and participants play correctly, then you will have an idea of how risk aversive a participant is, indicating how they might act (or did act) in the biosecurity game. At the very least, it can help explain the actions a participant made in the biosecurity game.

## Biosecurity Game

In the Biosecurity Game, there is a total of 15 rounds, each round participants will decide their biosecurity effort against biosecurity threats to prevent outbreaks and to gain a profit from each round. In this section, I will explain the nature of how the game calculates the probability of an outbreak and uses a random function to decide whether an outbreak occurs, which will result in the participant receiving a profit or loss. I will also go through how the biosecurity game uses certain JavaScript elements for the sliders present on the website. Now there are a few defaults that are existent across every session configuration, these defaults are as follows:

* The maximum amount of protection (or biosecurity effort) someone can offer against biosecurity threats per round is $10. This default will be labelled as **MP**.
* The minimum probability that a participant isn’t the source of the outbreak is 60% (0.6). This will be labelled as **PC**.
* The revenue for each round when no outbreak occurs is $25. Participants will receive no revenue when an outbreak occurs as the whole round’s crop is lost. This will be labelled with **R**.
* The upkeep, or the cost of production for crops per round is $5. This will be labelled with **U**.
* The starting funds for each participant is $50.
* The calculator showing a participant’s probability of them not being the source of the outbreak is always available.
* A participant automatically receives the equivalent of $10 in their local currency for playing the game (exceptions can be made at your discretion, depending on your moral and ethical obligations)
* The conversion rate from the game’s monetary amounts to the local currency is 0.1.
* The players per group is 4.

The point of the Biosecurity Game is to see the psychological, economical, and biological effects of biosecurity, focusing more on the psychological and economical. The psychological effects under the microscope are the group’s trust of each other and how people act (and react) when participating in programs devoted to biosecurity, where the risk of outbreak is relatively high. The economic factors come into it, when we analyse the patterns that exist when participants play the game, to see if they put themselves in a Nash equilibrium, or consider how other participants will act, and adapt based on this educated guesses or information. Morality factors do come into the game as well, but the effects one’s morality has on the game is investigated in the questionnaire after the biosecurity game.

Any biological effects can be inferred from the number of outbreaks in each game, and administrators could use this to possibly anticipate the severity of the impact of agricultural industries not properly following biosecurity laws. Although, the experiment alone will only take a small part in looking at biological factors, only supporting any conclusions on biological effects from certain biosecurity threats, and the how agriculture could affect its environment.

Now to explain, how an outbreak is determined, and how a participant receives a profit or loss.

### **How an Outbreak is Determined**

C = Amount of protection a participant put in that round ($0 ≤ P ≤ MP)  
  
P1, P2, P3, P4 = The probability that a participant (4 participants in this scenario) is NOT the source of the outbreak (0.6 ≤ P1, P2, P3, P4 ≤ 0.9995). Will be denoted as **Px** when referring to the function for P1-4.

**RAND =** A random number between 0 and 1 generated by the server  
   
1) Get the Cost Factor:

**CF =**

2)Determine the probability that a participant is the source of the outbreak:

**Px =**3)Determine the probability of no outbreak for the group (the set of participants containing P1­ – P4):

**Probability of No Outbreak (PNO) = P1** x**P2** x**P3** x**P4**

4) Determine the probability of an outbreak for the group:

**Probability of an Outbreak (PO) = 1 – PNO**

5) Now generate a random number between 0 and 1 and follow the inequalities below:

**if PO > RAND, Outbreak Occurs**

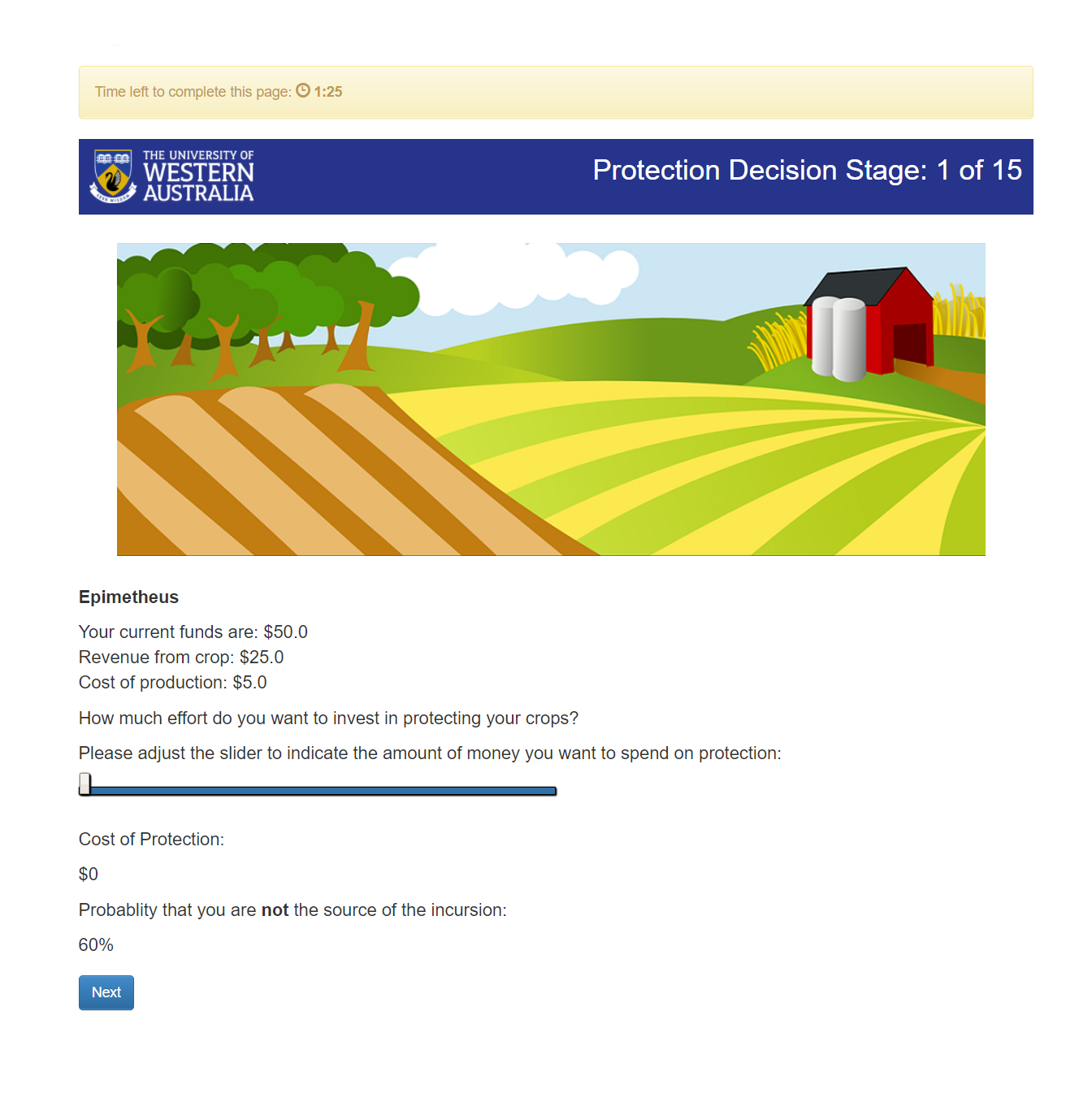
**if PO < RAND, No Outbreak Occurs**

6) Now determine each participant’s profits or losses as per the following functions:

If there was an Outbreak, then: **Current Funds = Current Funds – C – U**

If there was no Outbreak, then: **Current Funds = Current Funds – C – U + R**

### JavaScript Elements

In the biosecurity game, sliders appear during various stages depending on the session configuration. The slider I want to discuss is in the page where participants decide their biosecurity effort against biosecurity threats (and by extension, the individual pledge page which uses the same slider).

The slider above utilises JavaScript functions to dynamically change on-screen in front of participants. Namely, using an ‘onchange’ tag within HTML’s range input type, we call a function which simply changes the text on screen. The function that changes the text on screen retrieves the slider’s current value when a participant drags the cursor along the slider, and then uses the slider’s value as the **C** value in the equation from step 2 in [How an Outbreak is Determined](#_How_an_Outbreak). The JavaScript from here takes the output of the equation, multiplies it by 100 to present the output as a percentage, and then adjusts a HTML block denoted by a div tag, to display the output. The function also manages the presentation of the cost value displayed as ‘Cost of Protection’, displaying the slider’s value as a monetary value.

## Questionnaire and Results

The questionnaire that occurs after the game asks some questions about the participants experience during the biosecurity game, namely their motivation behind their actions and the trust of their group members. The questionnaire will also ask participants some questions that incorporate some thinking about morality, their answers to these questions can give an indicator how much their own morality impacted their decision making. For example, consider the 1st question asked in the questionnaire:

*Please Identify the most important factor underpinning your decision whether to protect or not during the biosecurity game?*

One of the possible answers to this includes ‘I thought that choosing to protect was the right thing to do, regardless of what other players decided’, a participant answering the question above with this answer can give you an indication that a participant made decisions based on their morality, rather than an idea of profit or loss. The Questionnaire does include questions about their gender, age, suburb, postcode and their annual income so an administrator like yourself can categorise results based on socio-economic categories.

The Questionnaire is generated from a CSV file inside the “CSV” directory within the project. The program reads the CSV file, taking values from the ‘question’ column to grab the question, and the values from the ‘choices’ columns to display the options each participant must choose from. The ‘type’ column, dictates the type of question it is, most are multiple-choice questions and hence are denoted using choice, others like the age or suburb are different and require distinct types. The lottery and biosecurity game control questionnaires that test a participant’s understanding of the instructions and the game, also work the same way, except for having an ‘#correct’ column inside the CSV files containing the questions. Since these questions have correct answers, we need to specify which answer is correct, the ‘#correct’ column contains integer values which dictates which choice is correct. For example, if ‘#correct’ has a value of 2, then the column ‘choice2’, contains the correct answer.

## How the Payoff is Calculated

Before the final questionnaire about a participant’s experience, there is a wait page that occurs for only one purpose, to adjust the payoff for each participant for the lottery and biosecurity game. It is here that one must understand how oTree calculates how it stores how much you’ll need to pay each participant. In oTree, each participant has a payoff that is associated with each application, this is stored within the player class of that application. To calculate how much you owe each participant, it adds all the payoffs inside the player class from all the applications, and stores this value inside a participant variable called ‘payoff’, which persists throughout each session. This ‘payoff’ participant variable is what gives your value inside the Payments Tab which will be explained here (add link 🡨). Please keep in mind that when looking at the Payments page, that the indicated monetary gain for the participant is most likely incorrect until this important wait page occurs.

The reason for this is due to the ethical obligations of the experiment for which this game was made. Thus, we had to apply the following rules to ensure participants earned money or didn’t lose money from the participation fee (the money a participant receives for participating in the experiment):

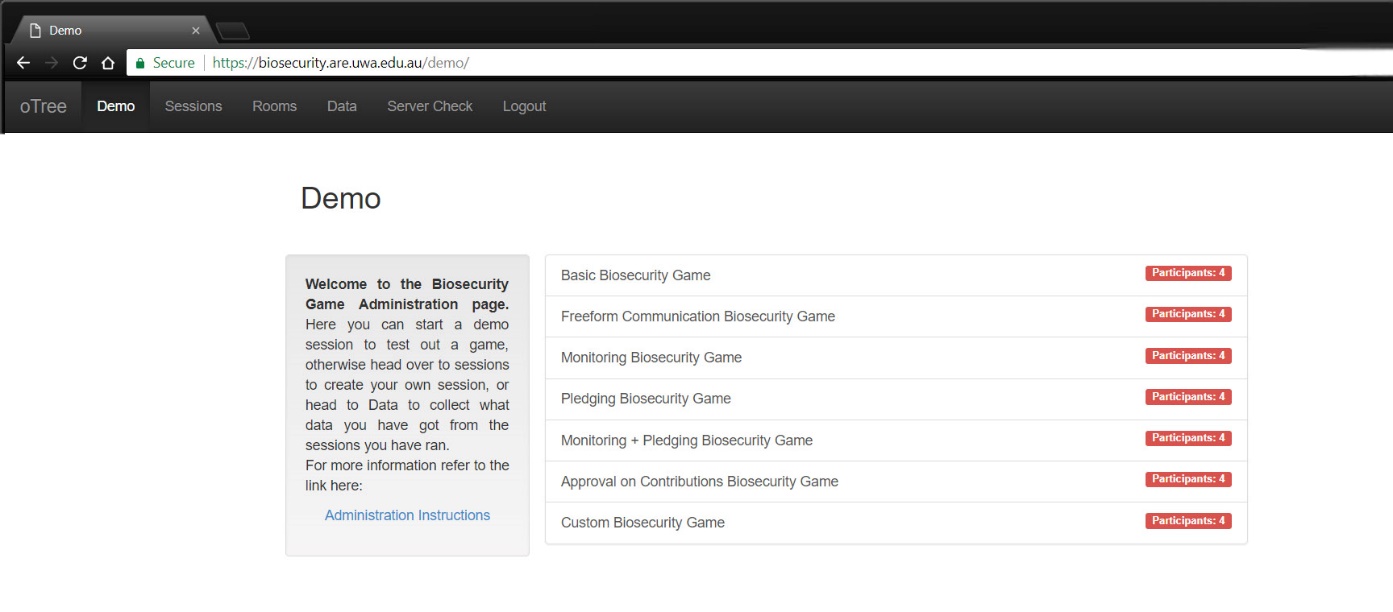
1. If a participant won money in the Lottery Game, but made a loss in the Biosecurity Game, then we add the absolute value of the loss in the Biosecurity Game, thereby cancelling out the loss from the Biosecurity Game, leaving the participant with only the gain from the Lottery Game.
2. If a participant made a loss in the Lottery Game, but made a profit in the Biosecurity Game, then one of two scenarios will occur:
   1. If the profit made from the Biosecurity Game is larger than the loss from the Lottery Game, then the loss from the Lottery Game is subtracted from the Biosecurity Game profit. This is the case where the payoff participant variable will show the correct result before and after the wait page, as a participant earns money.
   2. If the profit made from the Biosecurity Game is less than the loss from Lottery Game, then the total profit from both games will be $0, to ensure the participation fee isn’t affected.
3. If a participant made a loss in both the Biosecurity and Lottery Game, then the total profit from both games will be $0 to ensure the participation fee isn’t affected.

## Using the Website

While these sections will likely echo from information that can be found in the oTree documentation, I still feel the need to explain how to use the pages and what to expect when looking at the oTree administration site. I feel this is necessary as the documentation focuses more on the development of an oTree project (and perhaps rightly) forgoes the need to explain how to use each page, as each administration oTree site could look slightly different depending on the needs of developers and their clients or experiments. For each page, I will explain the purpose that each page serves alongside a labelled image explaining elements of the page. The combination of the image and explanation should tell you how to use the page and why you should use it. Most pages are intuitive to use, so don’t worry, learning to use the site should be easy for you, no matter your knowledge level.

### The Demo Page

The first thing you will see when loading biosecurity.are.uwa.edu.au is a login screen, the first text box will be the username, and the second text box will be a case sensitive password which as an administrator you should have access too. Once you have input correct credentials, you will be greeted with the Demo Page, it is here that you will see a list of the session configurations and welcome text (subject to change). Clicking on one of the session configurations will start a demo session, this demo session is volatile in the sense that once you close the tab/window in which the administration page of the demo resides, you will lose the ability to control the demo session, or see its results. This demo page is about testing the session configurations and experiencing the games (mostly) without consequence to the data that you are storing. You might want such features to show colleagues how the game works without having to start a session and configure it, or in the case of development, test an application or session configuration.



This is the **Main Menu**; the current page is highlighted in a darker colour.

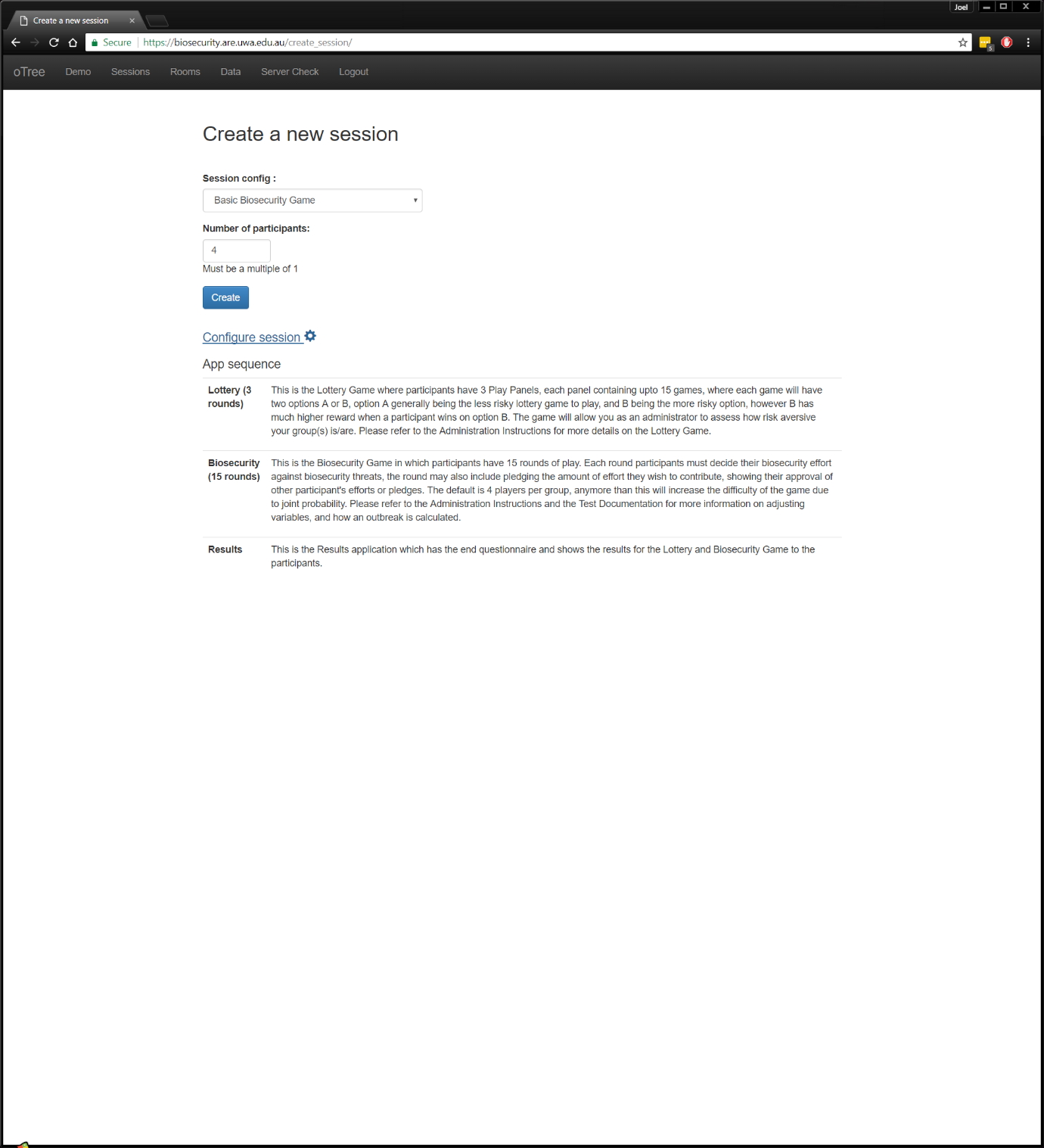
The welcome text (subject to change) introduces to the website and other documents you should be aware of, like the one you’re reading!

The list of session configurations shows the name of the session configurations and the number of participants a demo session will default to. For example, ‘Basic Biosecurity Game’ is a session configuration that will have 4 participants and run the entire game from Lottery to Biosecurity to the end Questionnaire.

The number of participants that the corresponding demo session will have.

### The Sessions Page

The sessions page will end up being the page where you as an administrator will spend most of your time, it’s on this page where you will create and administer sessions. Upon entering the page, you will see a list of your sessions that you have created, each could be in varying stages of completion. However, if you have no sessions then you’ll need to create one to start a session let’s start there. First, you’ll need to click the ‘Create New Session’ which will greet you with the two forms. The session config is a drop down menu where you will select the particular game you want to run a session with and remember that the number of participants is the number of participants in the session, not the number of participants per group. If you want more control over the variables used in the session and the game in particular then click ‘Configure session’ which will reveal the variables that you can change in order to change how the game works. For example you could configure the session to have more than 4 participants per group, set a higher revenue or change the minimum probability of no outbreak etc. For more detail refer to the annotated images below and on the next page.



The session config drop down menu, click here to reveal the different session configurations that are built in.

Type in the number of participants for your session.

Click here to configure the session further. See the next page for the explanation of the options.

The above will be the order that the session will run its applications. So, in this case, like in the game with Inigo it will run the Lottery Game, the Biosecurity Game then show the survey and the results of both games.

#### **Configuring a Session**

**In the order that you see them when configuring a session:**

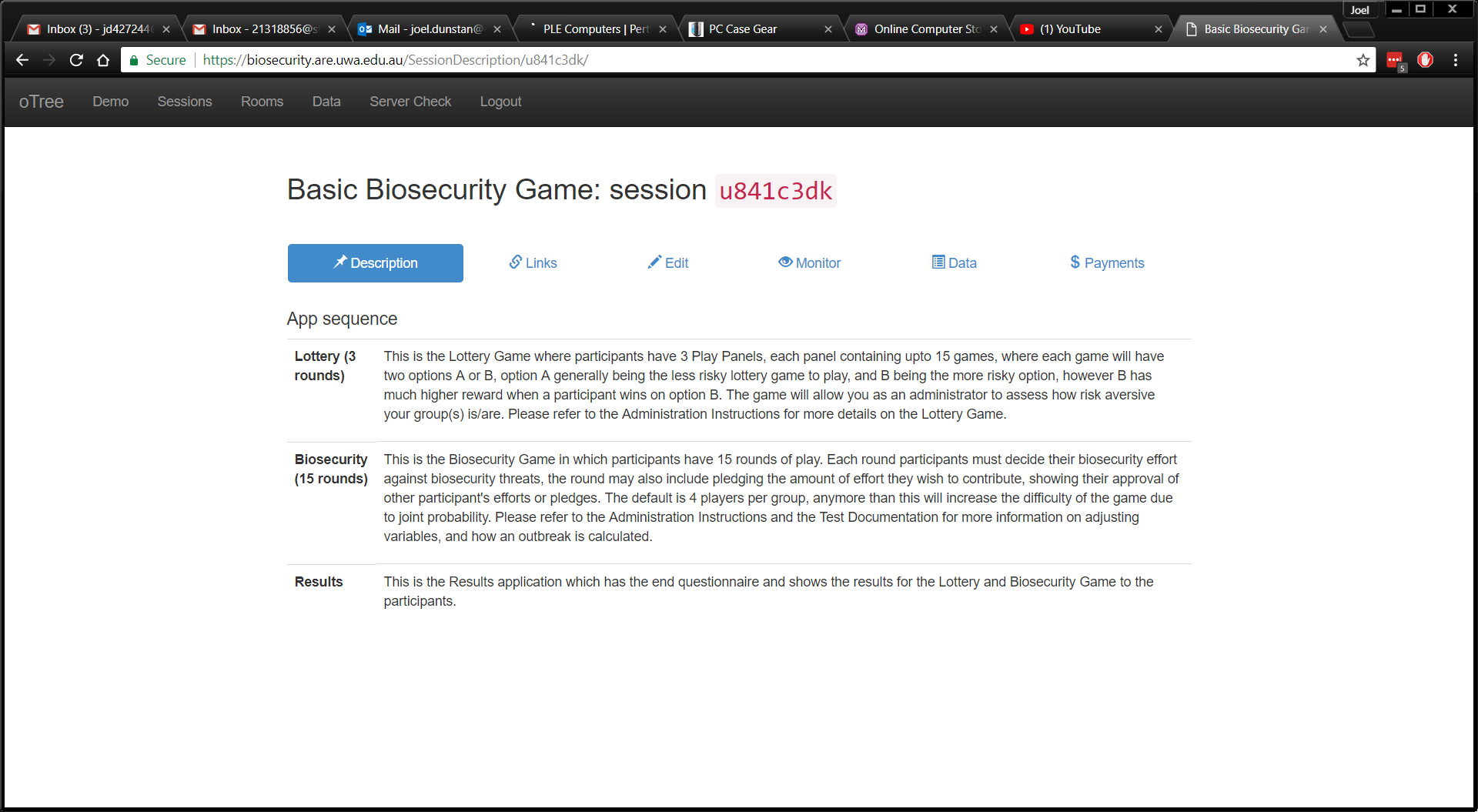
|  |  |
| --- | --- |
| **Variable Name** | **How it changes the game** |
| **Capproval** | Tick this box if you wish to enable approval by contribution. |
| **Papproval** | Tick this box if you wish to enable approval on pledges. |
| **calculator** | Ticked by default, turn this off if you don’t want participants to see the probability that they are not the source of the outbreak while deciding on their biosecurity effort |
| **case** | Ignore - Used for Testing |
| **Contribution\_looper** | This number determines which rounds approval by contribution take place when its enabled. If you put 3 then approval by contribution takes place every 3rd round. |
| **Controlling\_cases\_for\_bots** | Ignore - Used for Testing |
| **Dynamic\_finances** | Ignore – Deprecated and Untested |
| **Max\_protection** | Change this number in order to increase the maximum amount of effort participants can do during a round. Changing this will affect the game’s difficulty. |
| **Monitoring** | Tick this box if you wish participants to everyone’s biosecurity efforts during the Outcome Feedback Stage. |
| **Player\_communication** | Tick this box to enable the chat box to appear every 5 rounds. |
| **Players\_per\_group** | Change this number to affect the number of participants per group. Try keeping this to a number that is a factor of the number of participants in a session where possible |
| **Pledge** | Tick this box to enable pledging |
| **Pledge\_looper** | Change this number in order to change which rounds pledges occur |
| **Probability\_coefficient** | Change this number to change the minimum amount of probability that someone is not the source of the outbreak. Changing this will affect the game’s difficulty. |
| **Revenue** | Change this number to change the amount of revenue participants receive when there is no outbreak. Changing this will affect the difficulty of the game. |
| **Set\_leader** | Ignore – Deprecated and Untested |
| **Starting\_funds** | Change this number to change the amount participants start with at the beginning of the game. Changing this will affect the game’s difficulty. |
| **Upkeep** | Change this number to change the cost of production each round. Changing this will affect the game’s difficulty |
| **Use\_browser\_bots** | Tick this box to enable the browser bots which was used for testing. These bots will play the according to how I programmed them to play. |
| **Participation\_fee** | Change this number to change how much money the participants get by default, just by playing the game or participating in the session. |
| **Real\_world\_currency\_per\_point** | Changing this number will change the coversion rate between the in-game currency and real currency e.g. AUD = 0.01 x In-Game Currency. |

If the order has changed, I apologise but what the variables do shouldn’t change and you should still be able to use this as a reference.

### Running a session

Once you have created and configured your session, it’s time to run the session and administer its progress using 6 different pages available to you while running the session. These 6 pages are Description, Links, Edit, Monitor, Data and Payments.

#### **Description**

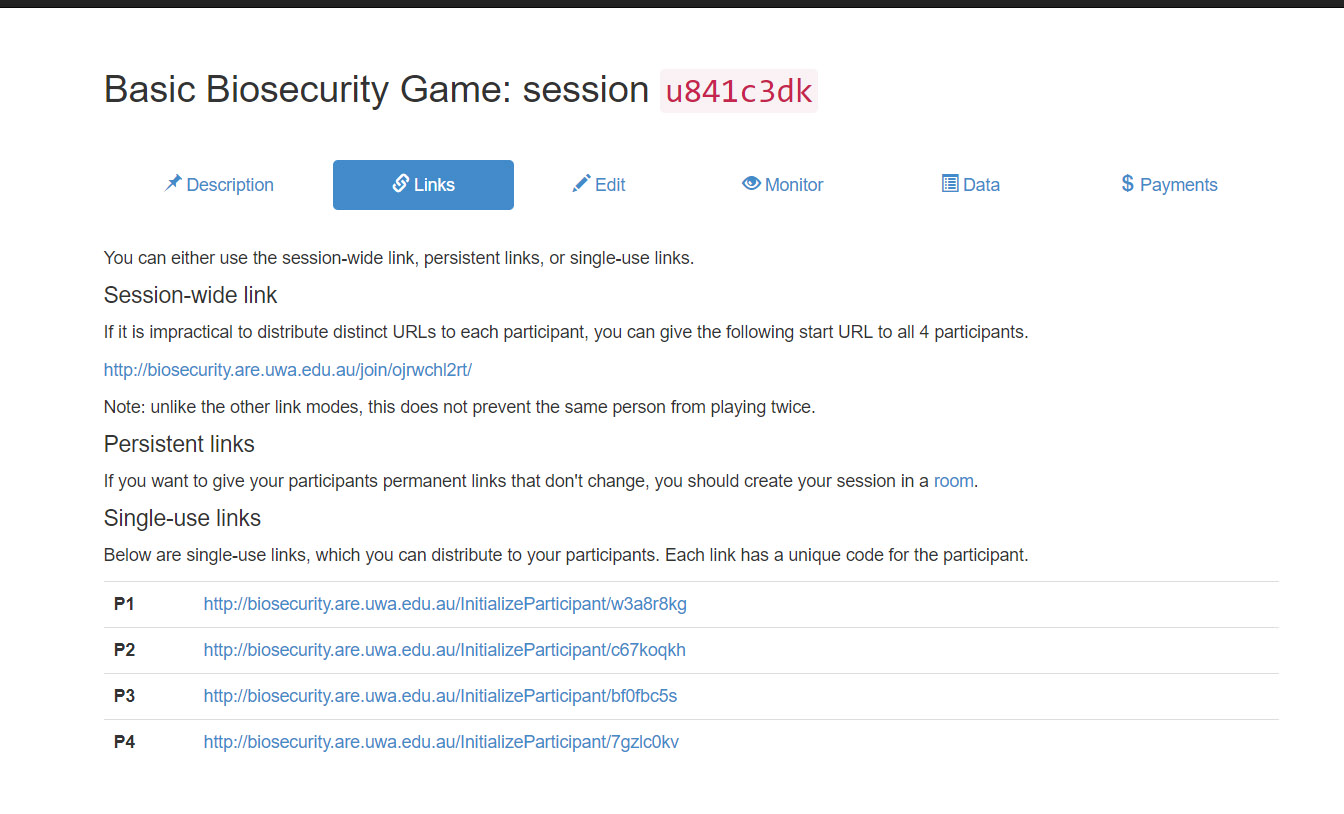
The description is simply a page showing the app sequence again, which also includes a description of each game. It’s not a particularly useful page, but its there if you ever need to remind yourself of the basics of the game, however it should be used in conjunction with this document.

#### **Links**

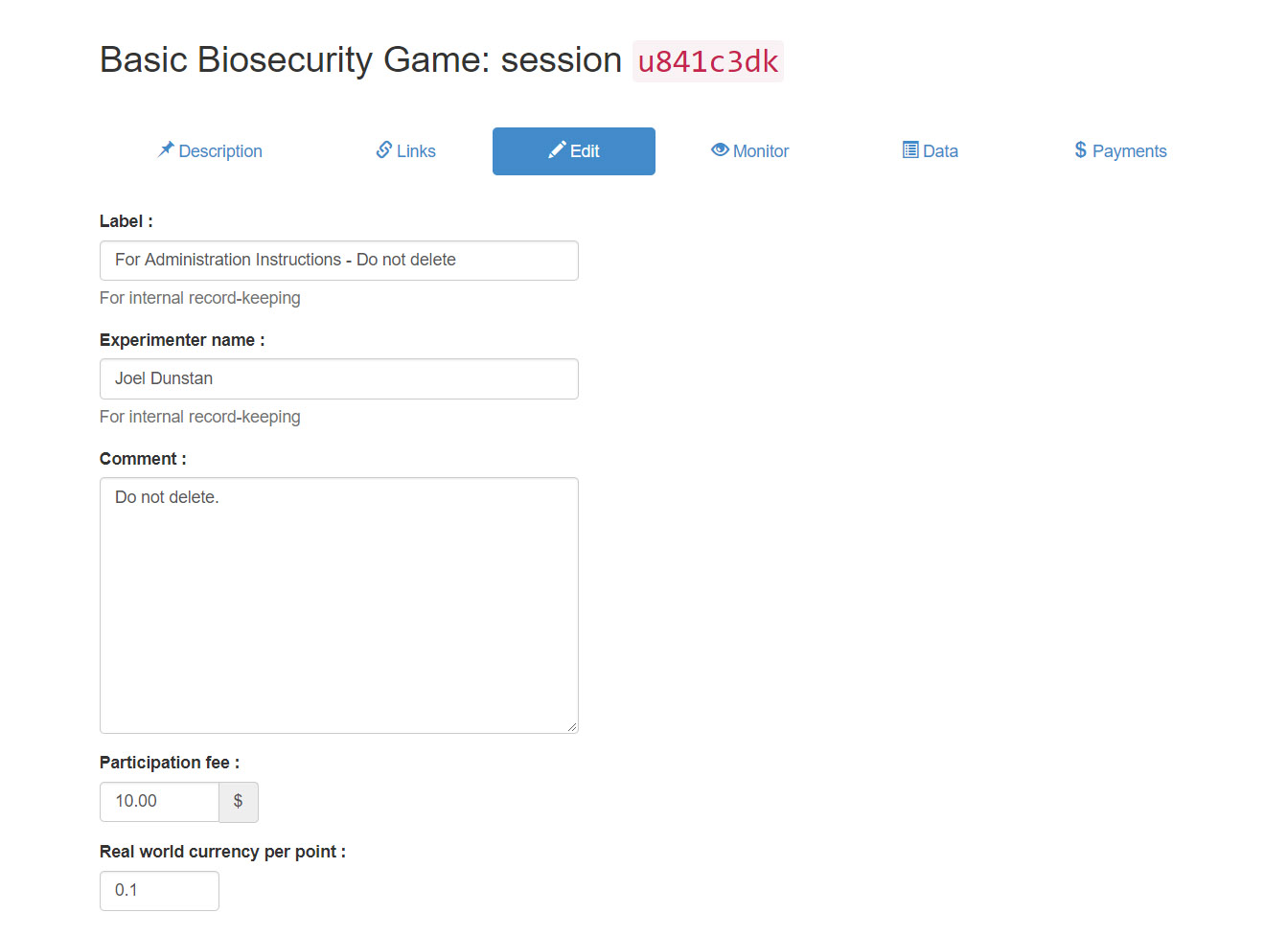
Once you have created a session, the first page it loads is the Links. The Links page contains all the links needed for participants to access the session and start playing the games. It is your job as an administrator to determine which type of links is best to use depending on where and how your session is being run. The two types of links I’m discussing here are the Session-Wide Links and Single-Use Links.

The Session-Wide Links are the easiest links to use as it allows to send each participant the same link. The upside of this is obvious, as an administrator it makes sending out the link(s) to the participants easy, you could send out a group email to all the participants, or if they are all in one room, write down the link on the whiteboard for participants to navigate to. However, the easiness comes at a cost, you run the risk of having participants clicking the link twice, meaning they could be two participants without your knowledge or perhaps every participant in the absolute worst case. Even if a participant accidently clicks the link twice, it unfortunately could lock one of your other participants out from the session, as the session-wide link can only be clicked a *n* number of times, *n* being the number of participants in the session. After the *n*th time, the link will simply lead to a page that says, “The maximum number of participants for this session has been exceeded.”.

If you decide the using a Session-Wide link isn’t practical, then you can use a Single-Use link. A Single-Use link is a link that is unique to each participant, so when a participant clicks this link, this directs them to their session and only their session, even if they were to click it twice, it would only direct them back to their current page within the session. This ensures that participants cannot be two participants without having another one of the participant’s Single-Use link. However, its less risky approach has a downside, as an administrator you will need to send *n* emails/messages, each email/message containing a unique link for each participant, meaning it will take longer for you to organise and start a session.

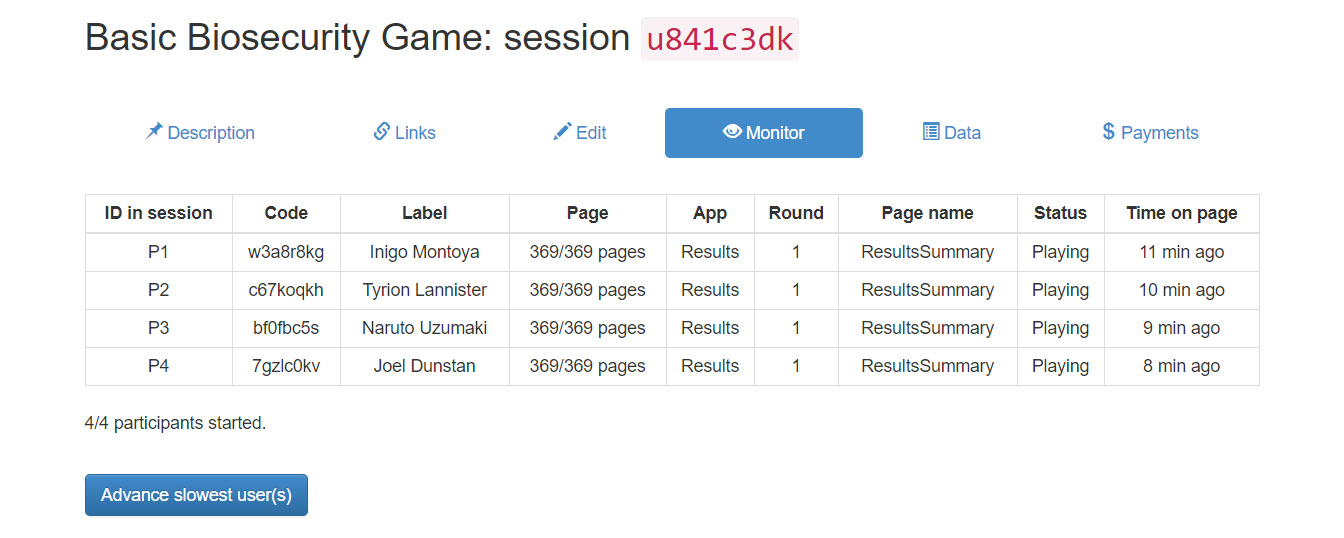
It is my recommendation you try to use Single-Use Links when it is feasible to do so, as improper use of the Session-Wide link will result in you sending a Single-Use link to fix the issue. There is another type of link you could use, I discuss Persistent Links here: [Rooms](#_Rooms).

#### **Edit**

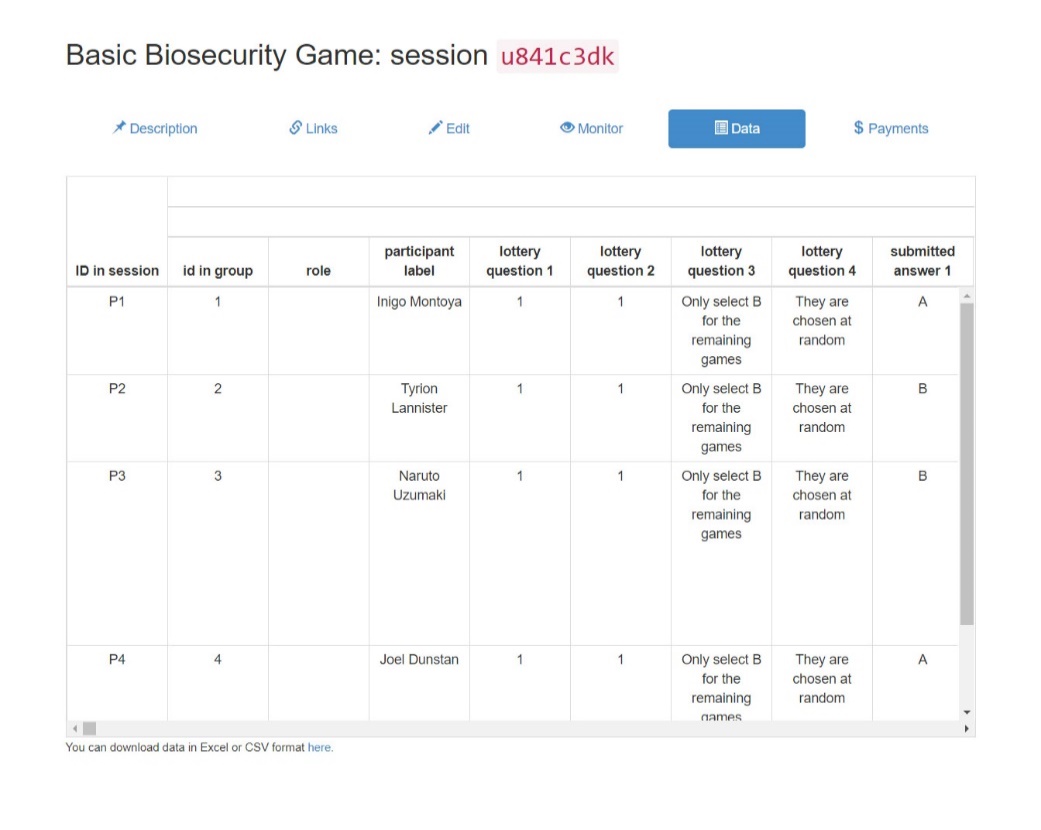
In this page, you can edit parts of the session relating to the record keeping of the session so its easier to find the session later in the website, particularly if you’re running multiple sessions in the same day/week/time. To make it easier to discern one session from another, you can add a label, the experimenter name and a comment to the session. To do this simply add text into the blank (by default) text boxes and click next once you’re done. My suggestion for label names include the name of the session configuration you’re using and the date and time, and for the experimenter-name put yourself. For example, “Basic Biosecurity Game 2 – 2 – 2020 14:00” for the label name and “Joel Dunstan” as the experimenter name. Another example I did is below.

#### **Monitor**

The monitor page is one of the ways to observe the progress of participants throughout the session and if necessary move especially slow participants up to the same round as other participants. The information this page shows in relation to a participant’s progress includes their participant label (which will be the participants actual name), the page they are currently on including the name of that page, and the time spent on the page. If you wish to move slower moving participants up you can click “Advance slowest user(s)”, however be warned clicking this will move *all* participants that are also on the same page as the slowest user, meaning you could affect multiple participants when using it. Generally, I have found that using this button wasn’t needed as participants would learn how the game worked quickly and moved through the game without any issues.



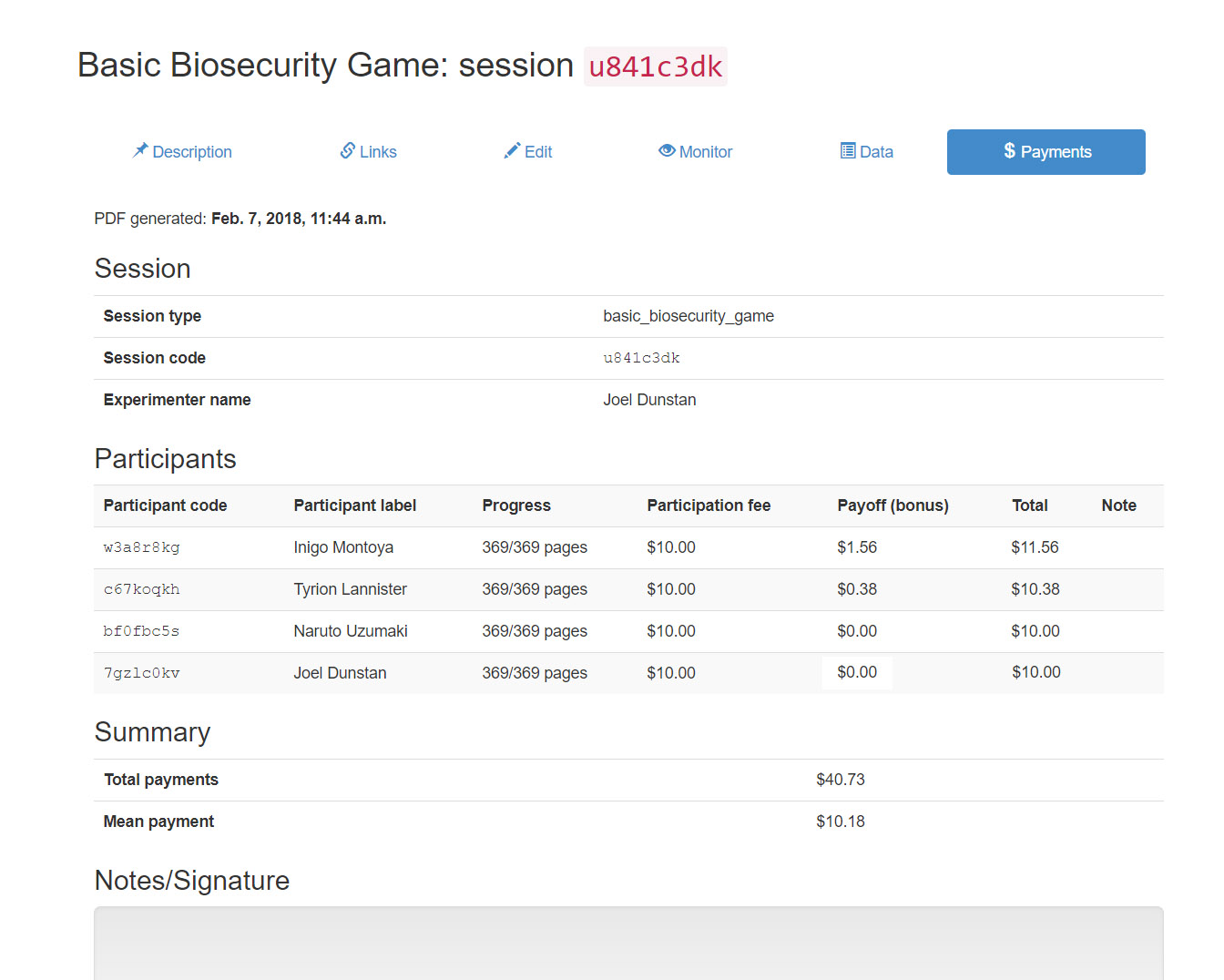
#### **Data**

The data page is another way to observe the participants progress throughout a session. As the name of the page suggests you will see the data the participants produce in real time. The data is recorded with each row representing the data from one participant, the data will be available even after the session is finished and will remain available until the session is deleted from the website.

#### **Payments**

The payments page displays the amount of money each player has accrued from the game in terms of the real-life currency you may or may not be paying the participants. Here each participant is displayed row by row, showing their name, current progress in the game, the participation fee and of course the payoff, the amount of money each participant has won. The total next to the payoff is the amount of money will you pay your participants. This page will also show you useful information, such as the total needed to pay all the participants and the mean payment of each participant. The payments page is also structured like an invoice, making it easy to print out and keep for records if need be.

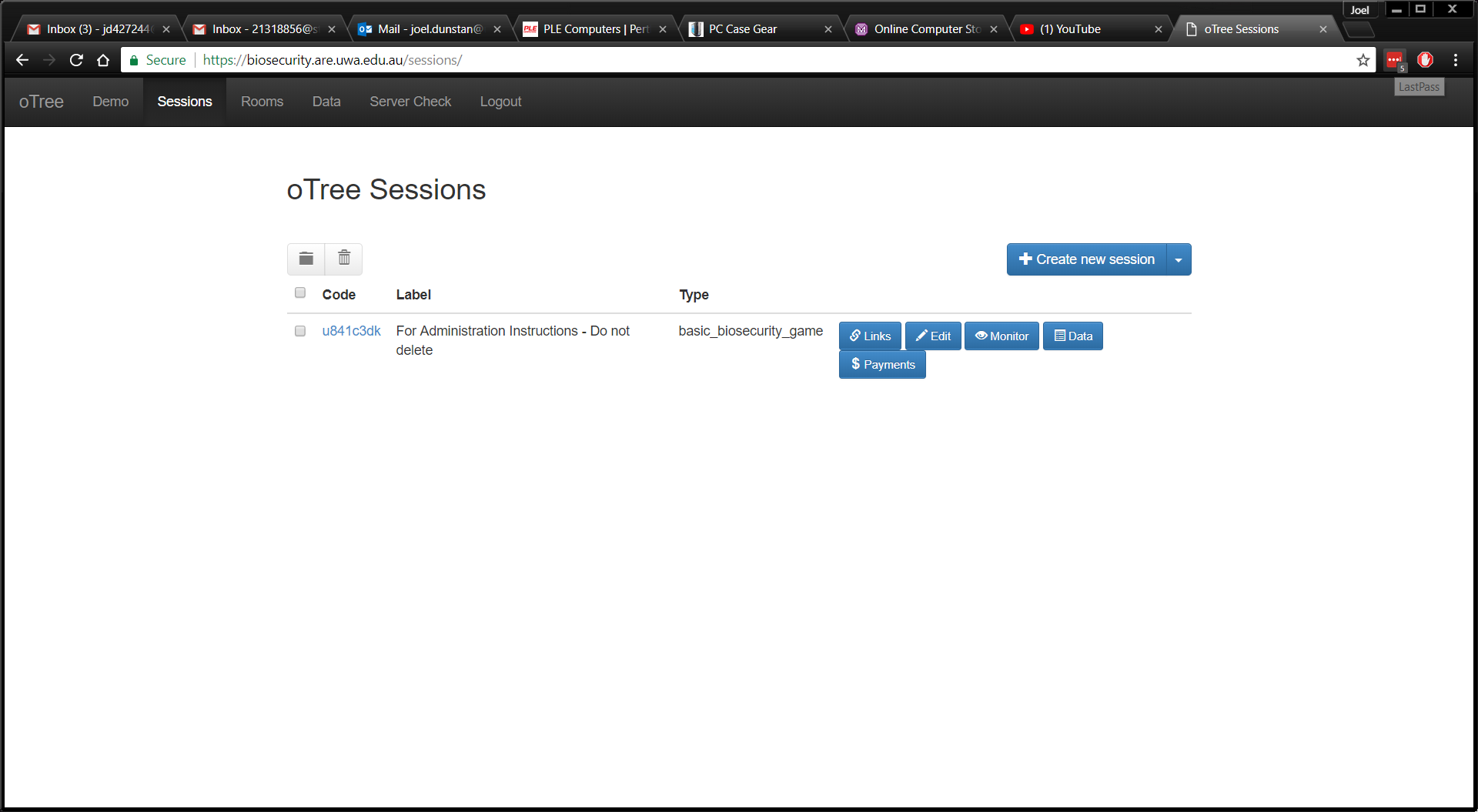
Keep in mind, that the values you see on the payments page won’t be correct until the session is finished due to how the payoff is calculated, see [How the Payoff is Calculated](#_How_the_Payoff) for more information. You can still look at the payments page while the session is in progress, just keep in mind that at that point, it’s unlikely that the payments page will be correctly showing the values you are paying the participants.



### Managing your Sessions: The Deletion and Archiving of Sessions

Once you have completed the session it is up to you as an administrator what you want to do with that session. You have 3 options:

1. You can just leave the session alone, and leave it in your list of sessions, making it easily accessible via the Sessions page. If you wish to do this, then I strongly suggest using the Edit page of each session to put a label for each session, so you can identify which session it is and when it ran. The downside to this approach is that after a while sessions will build up on your Sessions page making it harder to find sessions that you may have only just created let alone sessions you possibly made days or weeks ago. The upside is that the session will always appear in the data when you export the excel or csv sheet that contains the data.
2. You archive the session, to do this tick the check box next to the unique session code and then click on the folder icon which will archive the ticked session(s). If you wish to archive all non-archived sessions, then tick the check box next to the **“Code”** label, which will select all sessions and then click the folder icon. The upside to this approach is that the data for the session will always be accessible, but the session won’t be in constant view when you view the Sessions page. If you wish to see the session click “Show archived sessions” which will list all the archived sessions alongside your non-archived sessions. A possible downside in the huge amount of data that will stay in the excel or csv files when exporting the data as it will show the session data not just for non-archived sessions, but archived ones too.
3. You delete the session, to do this tick the check boxes for the sessions like you would if you were about to archive the sessions, however instead of clicking the folder icon, click the trash instead. It will ask if you wish to delete the sessions to prevent you from accidently deleting sessions. Deleting a session is inherently risky, the action is irreversible as its impossible to retrieve a deleted session. Before deleting you must ask yourself two questions before deleting the session, the answer should be yes to both before deleting:
   1. Do I have the data from the session or no longer need the data from session?
   2. I no longer need to use the session again for any purpose other than collecting the data from it, is this correct?

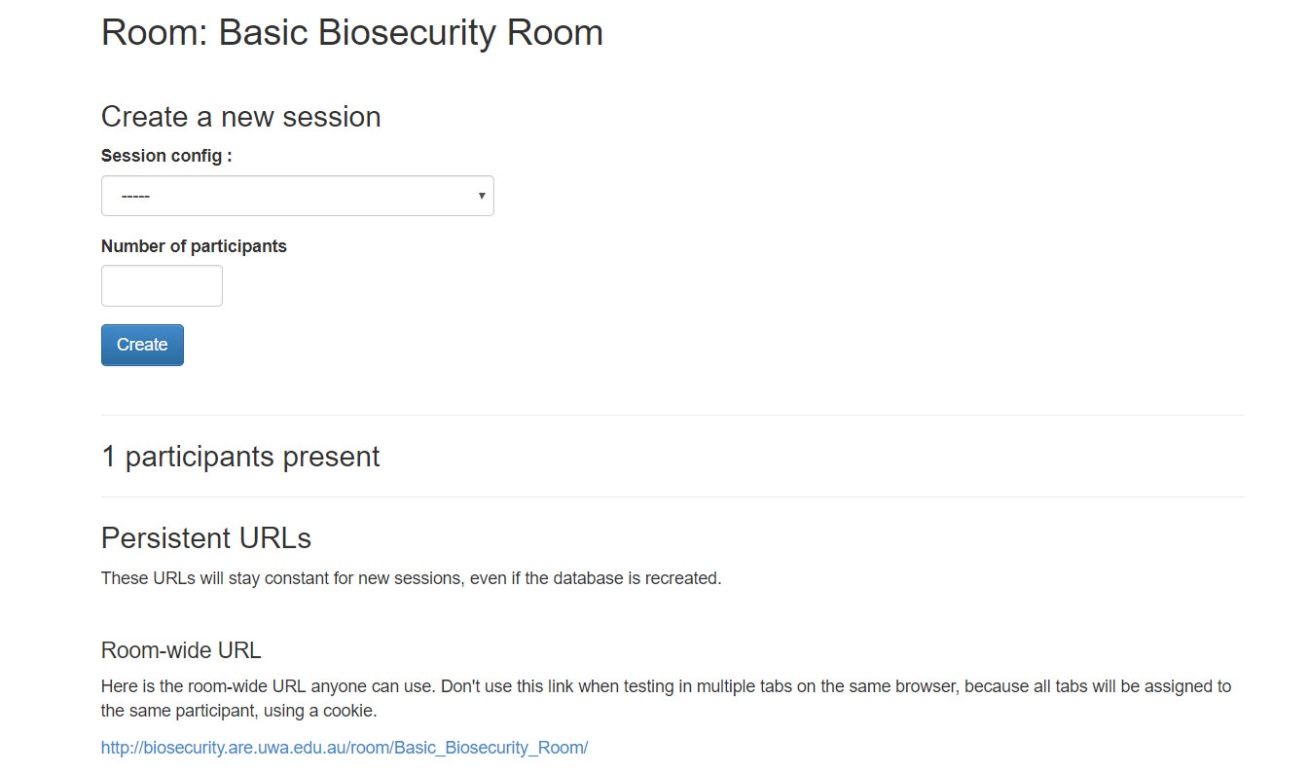
Deleting a session will remove the ability for data to be retrieved from the session, thus **deleting a session must be done carefully, not even I, nor the developer of oTree can retrieve a deleted session that hasn’t been backed up properly**.

### Rooms

The Rooms feature in oTree is another way of running a session, it exists to provide administrators an easier way of running sessions in a lab or classroom. I have created a ‘Basic Biosecurity Room’ for use, in this room you will have access to a persistent link. In comparison to the single-use and session-wide links (as discussed [here](#_Links)), persistent links in theory combine the best of both worlds. With persistent links you’re able to send one link to every participant like you can with session-wide links, however they come *mostly* without the risks of session-wide links, like single-use links they ensure that if someone were to click the link again it will redirect them to current page they are on in the session.

I say mostly without the risks of session-wide links as persistent links can be clicked twice by same person if they have more than one device. Persistent links uses cookies (metadata about your access to a website/server) to see whether a participant has clicked the link before or not, cookies are kept on devices that accessed the website, hence by checking the cookies the server checks whether the **device** has accessed the link before, not the participant specifically. This means, a participant with access to the link on two different devices could access the link on each device and join the session as two different participants. For example, I could be two different participants for one session as I could access the persistent link on my smart phone and desktop, thereby joining the session twice appearing to be two different participants when in-fact, I am one participant.

I still recommend using single-use links when its practical to do so, however if you’re administrating many participants in one session each with their own device to join the session in an environment where its unlikely participants will attempt to use two devices to play the game, then persistent links are quite useful.

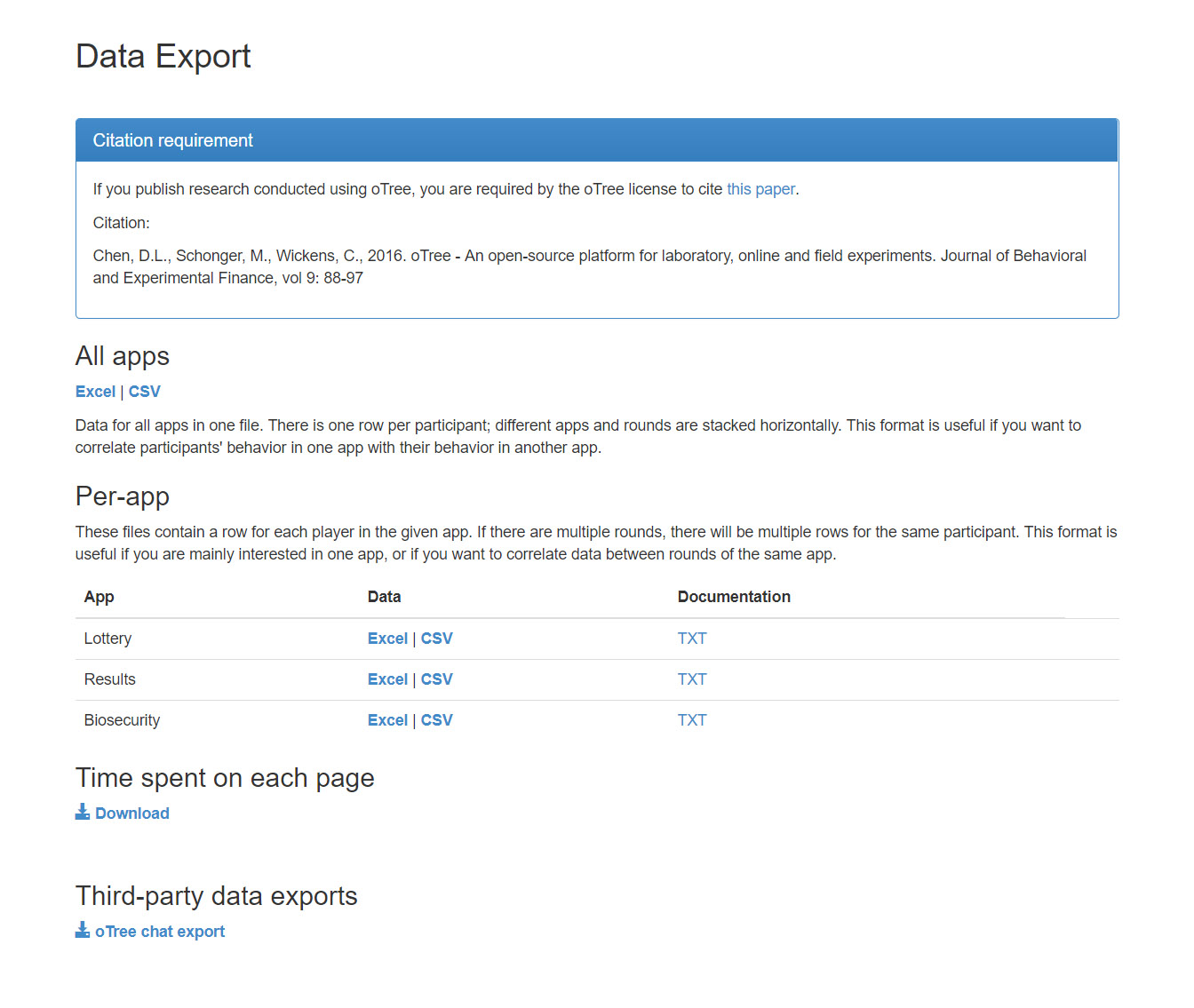
In terms of running a session with a room, you will need to click the link for ‘Basic Biosecurity Room’ in the list of rooms on the ‘Rooms’ page. In this room you will need to create a session with the number of participants being equal to the number of participants in the room being the best of use of the rooms feature. After this, it’s just like administrating a session as you would normally, except the room will be considered active until you close it thereby cutting the link between the room and the session allowing you to create a new session in that room. Sessions created in rooms will always be accessible via the Sessions page, even after the room is closed.

### Data Export

The data page is where you’ll access all the data your sessions have created, you can have the data in two file formats, csv or an excel file. Whether you download the data as a csv file or excel file, the data can be arranged in two ways. The first being via the ‘All apps’ links, the all apps links will arrange the data so that there is one row per participant, with each of the apps and the rounds within those apps are shown as a new column. This is the arrangement shown on the [Data](#_Data) page within each of the sessions you have/are administrating. For example, take our basic game with Inigo, referring to the excel file [here](https://1drv.ms/x/s!AphuEhKWv8YUg5tuT7myAyFE5630Ng), all data relating to Inigo can be found just by looking at the one row that Inigo’s name is on, particularly useful if you wish to compare Inigo’s behaviour in the Lottery app compared to the Biosecurity app.

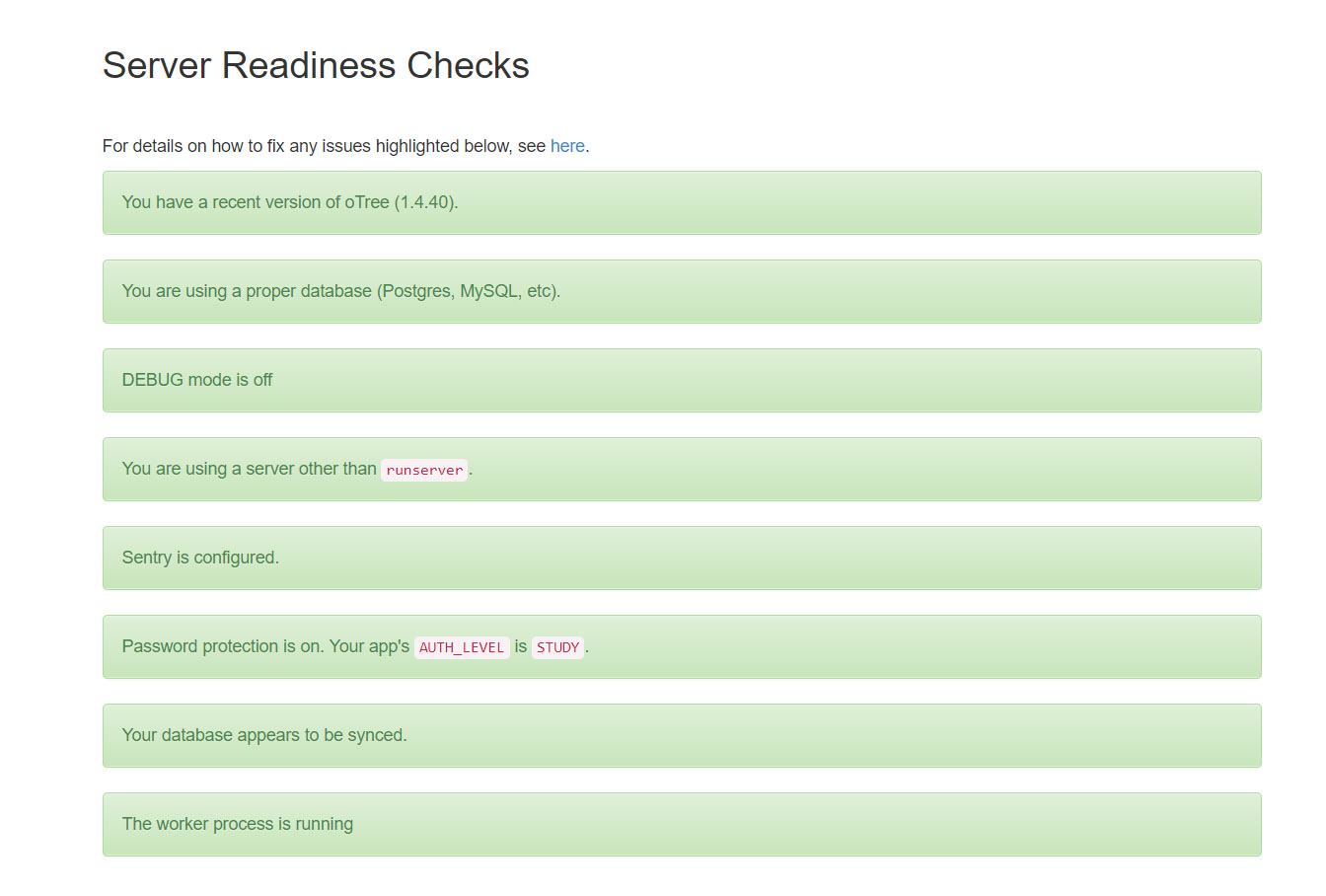
The other arrangement is via the ‘Per-app’ links, the per-apps links will arrange the data so there is one row per participant, per app, per round. For example, taking our game with Inigo again, referring to the per-app Biosecurity excel file [here](https://1drv.ms/x/s!AphuEhKWv8YUg5ttel0M01uJ-TeZLw), you’ll notice that all data referring to Inigo can be found through out 15 rows within the excel file due to there being 15 rounds in the Biosecurity Game. This type of arrangement is useful for looking at trends between the rounds within each app.

Other bits of data can be retrieved from the data page as well, including the time spent by each participant on every page, the logs of every chat that took place within a Freeform Communication Session Configuration and some of the documentation within the code via a text file. All these bits of data can provide you with all the information needed from the session once they have all been processed and observed to your heart’s content.



### Server Check

The server check page is all related to the server’s health. As an administrator you should check this page every time you’re about to create many sessions to ensure the server is ready for you to use. If there are any errors/issues, then the server check will highlight them in red. The only part of the server check that can be red is the first check which is related to the oTree version, since oTree is changing all the time and features/bits of code could become deprecated the server will only be upgraded when necessary. Thus, the version check may appear red when the oTree version isn’t the latest release, the server is currently on oTree version 1.4.40 which will (most likely) be forever the last version of oTree before its 2.0 release. All the other checks should always be highlighted green, if there are any that appear red with exception to the version check then please contact me [here](#_top). For more details on what each check is related to refer to the image below.



This check is about whether the worker process is running to allow people like yourself to access all the pages.

This check is about whether password protection has been enabled for the website.

These checks are about whether the appropriate error reporting has been enabled. DEBUG mode must be off, and Sentry activated.

This check refers to how oTree was started within the server, as oTree must be run a certain way when its being used outside of development.

This check is about the oTree version

These checks are about whether the database is functioning and is appropriate for oTree to use. I use Postgres the suggested DBMS for oTree.

# Being a Developer

Have any ideas and you wish to add to the game, or maybe you want to run the website on another server other than UWA? This section is about what you will need to learn to add apps or features to the Biosecurity Game, or to create a server ready to run the Biosecurity Game if need be. I will not be giving full tutorials or telling you exactly how oTree works, there are people out there that have done a better job than I could ever hope to. I will however try and give you some places to start learning so you can make the app/game whatever you envision it to be. I will try and give a step by step procedure on how to set up a server primarily on Linux, however I won’t be referring to a certain flavour as there are way too many to cover, however it is certainly easier on some flavours than others.

## Python and oTree

oTree is a python package that uses Django to run a multiplayer game designed to conduct behavioural, psychological, economic experiments or surveys/quizzes. oTree primarily runs through a Daphne server which uses HTTP and web sockets to run the website biosecurity.are.uwa.edu.au. oTree and Daphne together require Python v3.6+ to run the server and all the necessary packages that your oTree experiment depends on. Installing Python 3.6+ on Linux is easy! Some flavours of Linux already have the most recent version of Python installed, such flavours include those that are Debian based such as Ubuntu. If you do need to install them on Linux then try “**sudo apt-get python3-pip python3-dev”** if on Debian based systems, however systems like Fedora or CentOS which use ‘yum’, then you’ll need to do it the hard way, which is using a “**wget <link to python 3.6>”**, then extract it **“tar zxf <Python file>”**, then do navigate inside that extracted folder and then do a “**make”** and then a “**make install”**, alternatively if you don’t want it to replace the python 2.7 then do **“make altinstall”**. Please note depending on the flavour of Linux you have you may need to install some dependencies for Python before you can install Python 3.6.

On the other hand if you’re on windows then you’ll need to navigate to python.org and download Python 3.6.x using this link here: <https://www.python.org/ftp/python/3.6.4/python-3.6.4-webinstall.exe>. Follow the installation prompts and make sure that when it asks, you allow Python into your PATH, so you can run python commands from a terminal or command prompt.

If you ever need to check what version of python you have installed open a terminal/command prompt and type “**python -V**” which will return what version of python you have. You may have more than one version of python installed, as **“python -V”** may return version 2.7 while **“python3 -V”** or “**pip3 -V”** may return version 3.6. Ensure to always use python 3.6 if you have multiple python installations when using oTree.

Now that you know what version of Python you have, you should probably learn about it, the first stop you should take is here: <http://otree.readthedocs.io/en/latest/python.html>. In this page it goes over the basics of Python in terms of writing code, this page is good if you have knowledge of other programming languages (especially Object Orientated like Java) however if you’re starting from scratch your better off with a full tutorial. Check out for a great start: <https://docs.python.org/3/>, this is the official documentation on python. Another place to start is here <http://www.learnpython.org/> for a more interactive type of learning. You don’t have to look far for Python tutorials, even at your local news agency don’t be surprised if you find books about Python. For oTree you’ll only need Beginner to Intermediate knowledge of Python, and for the Biosecurity Game in particular, learn about the math, CSV and rounding packages inside Python as they are used extensively throughout the project.

Now for learning oTree, luckily Chris the developer of oTree has created documentation which goes over just about every aspect of oTree, he has some basic tutorials where you recreate some of the sample games that is provided in the oTree package. Chris in here also goes over the basics and gives an overview of oTree which will allow you to understand the structure of oTree and why I was careful with using the words like participant and player. To start your oTree experience start here: <http://otree.readthedocs.io/en/latest/index.html> and work your way through the menu on the left. If you have any questions that can’t be answered by the oTree documentation there is a forum to ask questions, the link for that is provided in the link I gave you above.

Finally, oTree uses HTML templates like most other websites in existence to show forms, text, tables etc. so getting yourself familiar with HTML and CSS will certainly help you understand everything going on in the template. We also use JavaScript for the dynamic parts of the template, we only use it in a basic capacity, but none the less it is there, again some basics of JavaScript will help too.

Once you feel ready to try your luck at editing the Biosecurity Game, contact me, I can provide you the source files and answer questions if you have any. Please read the documentation provided in the code in terms of comments before doing so. I hope this was enough to get yourself started and Good Luck!

## Setting up a Server

The following section will go through the basics of setting up a server, I will provide a step by step process for Linux. The Linux tutorial will mainly have commands from a Debian based system, however I will try to give you some parts which allow you to install a server on Fedora or CentOS and the like. Debian based Linux flavours are the easiest to set up the server on as getting all the required packages is easier than other systems.

### oTree Tags

First thing is that is common with any setup with this particular server is that we changed a single oTree file that needs to be replaced in the oTree source files every time a new installation of oTree takes place, this also includes when you update oTree. To do this simply copy the file named **“otree\_tags.py”** located here **“path/to/Biosecurity\_Project/Lottery/otree\_tags/”**, to a directory in the oTree source which is here **“path/to/Python36/Lib/site-packages/otree/templatetags”** and replace the file that exists in there by the same name. If you don’t do this functions in the Lottery app will not work as expected, nor will the oTree commands work with the server.

### Linux

This step-by step guide on setting up an oTree server for Linux will follow the same Linux Server Setup guide as provided by oTree with some minor differences and extra steps for certain things such as running SSL to run the website in HTTPS and things to be aware of for different flavours. The Linux Server Setup Guide for oTree is here: <http://otree.readthedocs.io/en/latest/server/ubuntu.html>, throughout this I will refer to this for certain steps as they are the same.

1. Make sure you have everything installed and where possible run the latest version of oTree in our case we’re using 1.4.40, but if you’re reading this you could have taken steps to move to oTree 2.0 either way, the steps should remain the same. So, the first thing you should do is put in this command:

**sudo apt-get python3-pip python3-dev libpq-dev postgresql postgresql-contrib redis-server git**

Now in this first step, you could not install git if you already have the source files and hence you won’t need it, I use Postgres as my DBMS however you could use MySQL instead for example as long as it’s a DBMS that can handle concurrent requests even when scale up to include thousands of users and supports Django. In an environment like Fedora, you might find this step to be one of your most difficult as chances are most of these repositories aren’t available to you, thus you’ll have to manually find each one, compile and install along with the dependencies they will undoubtedly rely on. Account for extra time if you’re doing so, hopefully things install without any issues. Also, when on Fedora or the like, make sure to add postgres and python to your PATH environment variable, doing so will make your life much easier.

1. If you’re running multiple python projects on this server it is heavily recommended to use a virtual environment, in fact doing so is considered best practice either way, this way your oTree server runs in separate environment to your python installation. Follow here: <http://otree.readthedocs.io/en/latest/server/ubuntu.html#create-a-virtualenv>
2. You’ll need to setup Postgres and the environment variable needed for it, the steps you need are here: <http://otree.readthedocs.io/en/latest/server/ubuntu.html#database-postgres>. After following the steps instead of exiting out of Postgres you also want to allow whatever user you created the authorisation to create databases, but you want to do this without making this user a superuser. To do this you need to do: **ALTER ROLE <username> WITH CREATEDB**. While granting all privileges to a database seems to do everything, it doesn’t allow the otree resetdb command to run as the user doesn’t have the authorisation to create a database hence, the command doesn’t recreate the database and halts.
3. Check you have everything and it’s running, check your python version, check if you have the source files, have you done the oTree Tags as mentioned above [here](#_oTree_Tags), make sure redis is running by following the step here:

<http://otree.readthedocs.io/en/latest/server/ubuntu.html#install-redis>

Finally check your otree version and run an otree resetdb and see if the command runs without issues. If the database resets, then test it out by doing an otree runprodserver which will run the server. If it runs then great, shut it down there’s more to do.

1. You will now need to set additional environment variables, so the that server doesn’t run in DEBUG mode, has a password etc. Like you did in step 3 with postgres you’ll need to add exports to the bashrc file.

**export OTREE\_ADMIN\_PASSWORD=<your password>**

**export OTREE\_PRODUCTION=1**

**export OTREE\_AUTH\_LEVEL=STUDY**

1. Now to ensure your oTree server will run on startup and will be treated as a service, its best to use a process control system like supervisor or circus, the steps for those are here:

<http://otree.readthedocs.io/en/latest/server/ubuntu.html#optional-process-control-system>

On my server where I had to run RedHat Enterprise V6 (RHEL6), neither of the process control systems would work despite the permissions and all that I entered I found another way to start at setup using init.d, even though this isn’t recommended as its old school it was the only thing that worked for me. In theory crontab will also do this, but not even that worked for me, safe to say my server setup was plagued by trouble.

1. Set up Sentry so that you can receive any errors from the oTree server if they do happen since debugging has been turned off, its important. To set one up refer to here:

<http://otree.readthedocs.io/en/latest/server/next_steps.html#logging-with-sentry>

1. Now finally run your server using the command otree runprodserver 8000 which will run the server on port 8000. To access this, you’ll need to access a web browser and go to 127.0.0.1:8000 and see if the oTree administration website loads. If it does, then excellent you have a running oTree server. All steps from here are optional, you could run your server on port 80 means it uses the standard HTTP protocol and therefor you could just access the oTree website by just putting in your domain name or IP address for your server. Of course, you would need to open port 80 where necessary and allow if through your firewall. On the other hand, you could take a more secure approach and run HTTPS which requires more setup, that’s what the following steps are for.
2. Download and set up apache 2.4+ as only 2.4 and above supports web sockets, this isn’t a concern in Debian based systems like Ubuntu which has apache 2.4+ installed by default, however other systems aren’t so lucky, and you will need to install them through a make and make install, again check dependencies I had to install APR, APR util and pcre for RHEL6. When configuring apache 2.4+ make sure you have ssl enabled and anything else you might need enabled e.g. load balancing. In my case I had to tell it to include the apr and pcre as well.
3. Now you’ll need to get some SSL certificates that have been signed by Certificate Authorities, there are many ways to do including Let’s Encrypt which a free service and comes recommended by many. To set up Apache you’ll need to get httpd.conf and edit a few things like the enabled modules if you haven’t enabled all the required modules, you’ll need to have a VirtualHost which redirects any port 80 traffic to https (port 443). Once you have done this you’ll need to adjust your ssl.conf file to include all the paths to the required certificates that you have obtained, I also advise that you turn off all protocols except TLSv1.2 and disable all SSL protocols and adjust the ciphers if need be. Finally in the ssl.conf file you’ll need to add a reverse proxy and a rewrite so once port 443 is accessed securely it shows the oTree site.

You’ll notice that my step 10) is very general and that’s on purpose, its something you should not be doing unless you understand the basics of cybersecurity and encryption for doing something wrong could leave your supposedly secure website, not so secure. If you need to test your configuration there are many SSL checkers that you can use such as [www.ssllabs.com](http://www.ssllabs.com). Each Apache configuration will be slightly different based upon your configuration but hopefully the 10th step will let you know what you might need to enable or disable or to add to your configuration to make it work.

### Windows

Setting up the server on a windows machine is like setting up a server in Linux just with less time spent in the command line and more time in GUI’s. Here’s the oTree docs link for setting up a windows server: <http://otree.readthedocs.io/en/latest/server/server-windows.html>.

For any additional steps refer to my Linux guide in the previous pages as the postgres setup can be applied to Windows or Linux, the same could be said for the apache setup and redis setup. I personally found Windows to be the easiest to set up a server next to Ubuntu and can recommend running it on windows.

# Resources

* Data Export Folder containing all results from the Basic Biosecurity Game: <https://1drv.ms/f/s!AphuEhKWv8YUg5tgPoY9g37dQh4pVg>
* All the images for this document came from here:  
  <https://1drv.ms/f/s!AphuEhKWv8YUg4pA6-YEPDpAzEB-DA>
* The instructions for the games themselves and any editing that took place:  
  <https://1drv.ms/f/s!AphuEhKWv8YUg4pDW3WWTZ2o5WkjOA>
* For those who are developing, here’s my test document:  
  <https://1drv.ms/w/s!AphuEhKWv8YUg4dMCbtpc2Jjxu2c2A>
* Original README.txt for the experiment back in 2016:  
  <https://1drv.ms/t/s!AphuEhKWv8YUgtsn94edLiRnhS2UNQ>

# Acknowledgements

I’d like to thank Ben White and Mark Hurlstone for the wonderful opportunity of creating this game as it is today, I’d also like to thank my previous team members who with me developed the initial version of the game and gave me the base to build upon. Thank you, William Meyer, Trae Shaw, Campbell Beck, Martin Porebski and Justin Chong. I’d also like to thank Chris for providing support from the oTree development team, he was there to answer questions and there when I needed him.

# Glossary

* oTree: “oTree is a framework based on Python and Django that lets you build:
  + Multiplayer strategy games, like the prisoner’s dilemma, public goods game, and auctions
  + Controlled behavioural experiments in economics, psychology, and related fields
  + Surveys and quizzes” (oTree, 2016)
* Python: A high-level programming language that allows you to design and implement computer programs on any machine.
* Django: A web framework written in Python used to create complex, database driven websites.
* HTML: Stands for **H**yper**T**ext **M**arkup **L**anguage, it’s the language used to produce text, images, colour, graphics and links on the world-wide web. Every single page you access on the internet uses HTML.
* Web Server: The computer/device/machine that runs the necessary programs and services to allow people to access the website on the internet and run the oTree project using Python and Django on a web browser without a client’s computer needing Python, Django or any of the necessary programs to run the game. The term web server is also used when talking about the services or programs that deal with the input and output through HTTP, e.g. Daphne Web Server.
* Web Browser or Browser: An application that is used to retrieve and present information and graphics sent by a web server.
* CSS: Stands for **C**ascading **S**tyle **S**heet, it is a language used to format how HTML elements will appear on a web browser.
* Relational Database: A database that properly recognises many to many relationships and is structured in a way to store information efficiently and effectively.
* Postgres: A Database Management System (DBMS), to which a DBMS allows one to create and manage databases.
* HTTP: Stands for **H**yper**T**ext **T**ransfer **P**rotocol, a high-level protocol designed to take and receive requests for HTML pages on the internet, it is a major part of communicating on the world-wide web.
* Operating System: Often used as an acronym **OS**, low-level software that allows a computer to manage basic functions like scheduling tasks, providing API’s for other programs etc. Windows, Android, OSX and iOS are examples of operating systems.
* IP Address: Stands for **I**nternet **P**rotocol **Address**, it is your “location” in a network, you can think of it like your home address, a unique set of numbers (and letters if using IPv6) that identifies your computer.
* Hardware: A physical part, component of a computer
* Software: A virtual program that can run a computer/device/machine created using a programming language, and designed for a certain purpose.
* Session: In the context of oTree, “a session is an event during which multiple participants take part in a series of tasks or games.” (oTree, 2016)
* Protection: In the context of the game, this is the amount of effort a participant contributes during a round in the biosecurity game in terms of probability of the participant **not** being the source of the outbreak. This is not to be confused with Cost which is still the amount of effort a participant contributes, but represented in dollars.
* Outbreak: In the context of the game, this is an event when pests have infected the participant’s crop for that round and therefore get no revenue.
* GitHub or Git – A version control system that tracks any changes made to a piece of software from one person or amongst a group to coordinate and organise themselves on how to keep changing the software, if it needs changing.
* Repository – Git uses repositories as sources of code, the repository is the main source of the code to which a group will coordinate from, as Git stores the information about the project such as changes made to the code.
* Runworker – These are processes that are run when executing the commands for running an oTree server. Each process appears as a terminal on the desktop, they deal with the navigating and loading of the pages.
* Play Panel – This is a page from the Lottery application, in the page participants choose their options for the Lottery.
* oTree Project – An oTree project is made up of oTree applications, the applications are the games themselves, the project directory contains all the directories for the applications and the settings.py file which dictates all the session configurations and the overall settings to be used for the Web Server that will be run from the project directory. Such settings include the use of Sentry, password protection etc. Throughout the document, I use the word **game** to represent the **oTree Project or an application**. In the context where I only talk about one application, game refers to that application, in all other areas, game refers to the whole project.

# References

Chen, D., Schonger, M., & Wickens, C. (2016). oTree - An open-source platform for laboratory, online and field experiments. *Journal of Behavioral and Experimental Finance, 9*, 88-97.

oTree. (2017, May). *oTree Documentation*. Retrieved April 2016, from http://otree.readthedocs.io/en/latest/

PCMag. (2017, May). *HTTP Definition from PC Magazine Encyclopedia*. Retrieved from PCMag: http://www.pcmag.com/encyclopedia/term/44501/http