



University of Glasgow | School of Computing Science

Gold Digger: a searching behaviour game

Gabriele Giordano Maria Rossi

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

A dissertation presented in part fulfilment of the requirements of
the Degree of Master of Science at The University of Glasgow

Date of submission placed here

Abstract

abstract goes here

Education Use Consent

I hereby give my permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name: _____ Signature: _____

Acknowledgements

acknowledgements go here

Contents

1	Introduction	7
1.1	Overview	7
1.1.1	Problem statement	7
1.1.2	The metaphor	7
2	Survey	8
2.1	Background theories	8
2.1.1	Optimal Foraging Theory	8
2.1.2	Information Foraging Theory	9
2.2	Previous studies	14
2.2.1	Searching behaviour	14
2.2.2	Gamification	14
3	Design	15
3.1	Site Map	15
3.1.1	Home, Registration and Login	16
3.1.2	World Map	17
3.1.3	Game Screen	18
3.1.4	General Store	19
3.1.5	Leaderboards	20
3.1.6	Tutorial	21
3.1.7	Achievements	21
3.1.8	About Page	22

3.2	Walkthrough	22
3.3	System Architechture	25
3.4	ER Model	25
3.5	Graphics	27
4	Impementation	29
4.1	Development Methods	29
4.2	Heuristic Evaluation	29
4.3	Testing	29
4.3.1	Unit Testing	29
4.3.2	Live Testing	29
4.4	Technologies	29
4.4.1	Django	29
4.4.2	Python	31
4.4.3	HTML5	31
4.4.4	CSS3	31
4.4.5	Twitter Bootstrap	31
4.4.6	Javascript and JQuery	31
4.4.7	AJAX	31
4.4.8	Code	31
5	Results and Data	32
5.1	Data Logged	32
5.2	Data Analysis	32
5.3	Results	32
5.4	Reflection	32
A	First appendix	33
A.1	Section of first appendix	33
B	Second appendix	34

List of Figures

2.1 frequency of clams in class size (% on y axis) by clam length (x axis)	8
2.3 An energy gain function	9
2.4 The optimal rate of gain	10
2.6 (a) - The Graph illustrates Charnov's Marginal Value Theorem where t^* denotes the optimal amount of time spent within-patch given the intersection between the gain function $g(t_w)$ and the tangent passing from t_B (the time spent between patches)	13
2.7 (b) - Shows the changes generated by the adoption of between-patch enrichment strategies. The tangent to the gain function $g(t_w)$ passing from t_{B2} determines that the optimal amount of time to spend in a given patch is now t_2^* . The forager is now able to afford to spend less time in each patch.	13
2.8 (c) - Shows the changes generated by the adoption of within-patch enrichment strategies. The new gain function $g_2(t_w)$ has a higher rate gain which allows foragers that spend the same amount of time between patches t_B will be able to reap more results in a shorter time.	13
3.1 Gold Digger nav bar	15
3.2 Home Page	16
3.3 Registration modal	17
3.4 World Map	17
3.5 California modal	18
3.6 Mine	18
3.7 General Store	19
3.8 Leaderboards	20
3.9 Tutorial	21
3.10 Achievements display	22
3.11 Special Achievements	23

3.16 The Gold Digger ER diagram 26

Chapter 1

Introduction

1.1 Overview

1.1.1 Problem statement

This project proposes to analyse user's searching behaviour in a context deprived of any cues that could allow them to exploit any previous experience in the task in order to achieve an optimal information foraging behaviour. User's performance will be recorded and evaluated to see if it matches data in experiments in which this context is present. Findings could help provide insight on people's choices when faced with a task that requires the same kind of skills that are required in information foraging with none of the context that they are presented with in experiments based on the same theories. Finally this project aims at making user's experience as enjoyable as possible for two reasons. Firstly it will avoid users feeling like they are performing work, removing them further from a standard information foraging task. Secondly, in order to enhance the number of users that play the game as well as the amount of games played.

1.1.2 The metaphore

Chapter 2

Survey

2.1 Background theories

2.1.1 Optimal Foraging Theory

One of the most interesting studies proposed by Sineviro which can help us bring out some interesting points on searching behaviours, is the one on crows foraging on clams. In this study, Sineviro tells us about the foraging behaviour of the common crow (*Corvus caurinus*) in the intertidal. This kind of crow, has developed a technique to open clams which requires it to take a short flight and drop clams on some rocks below it to crack them open. The cost (in energy) of searching for clams, and the one of handling them is almost the same, however, the cost (in time) of performing the same activities is 4 times more expensive with regards to searching. It is then difficult to understand why crows would reject a large amounts of the clams they find, especially given the fact that searching seems to take up so much time. The reason for this behaviour is to be found in the "average net profitability of the clams as a function of size" (Sineviro 2006), which can be represented by the equation:

$$\frac{\text{Energy}}{\text{Time}} = \frac{\text{Energy per clam as a function of size} - (\text{Search Cost} + \text{Handling Cost})}{(\text{Search Time} + \text{Handling Time})}$$

It is very interesting to notice that predictions on the crow's behaviour in order to maximise energy gain per unit of time according to this formula, closely match its observed behaviour.

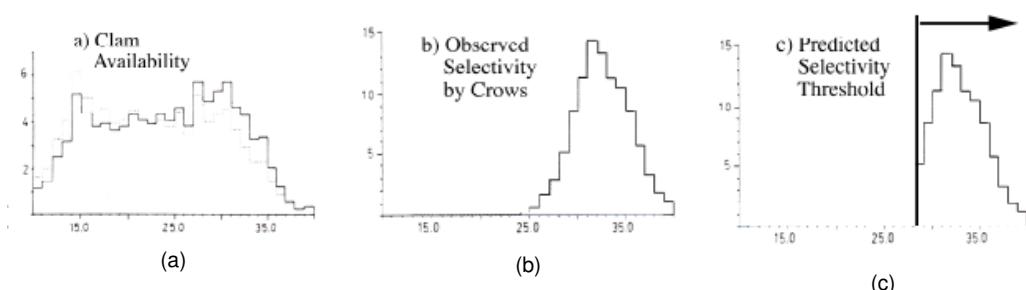


Figure 2.1: frequency of clams in class size (% on y axis) by clam length (x axis)

As we can see from (b) and (c) in fig 2.1, predicted behaviour seems to closely match observed behaviour. This kind of model, however, considers the disposition of prey in the environment to

be random but, sometimes, prey items seem to have a patchy distribution (Sineviro 2006). In this case, an animal will have to travel between patches before it can start exploiting a new one. This creates two variables according to which an animal has to "make its decision" in order to maximise its rate of gain. The first one is the time spent within a patch to feed, and the second one is the time spent travelling between patches in which, crucially, the animal doesn't gain any energy and instead consumes it. A formula that effectively models this choice is:

$$\text{Rate of energy gain} = \frac{\text{Energy}}{\text{Time}} = \frac{\text{Energy or Load Size}}{(\text{Travel time to patch} + \text{Foraging time to patch})}$$

Also known as "**Charnov's Marginal Value Theorem**" which generates an energy gain function like the one on the left. Here, we can see how there are two main areas delimiting the Cartesian space, the first one (on the left side of the curve) is the potential time the animal has to spend in between patches, whereas the second one is the potential time spent feeding in a single patch. An optimal allocation of time in foraging for this rate of gain will be represented by the tangent to the curve as shown in the graphs on the right. The red line is the tangent and the point of intersection with the curve determines the in-patch time that would be optimal for an animal to spend feeding.

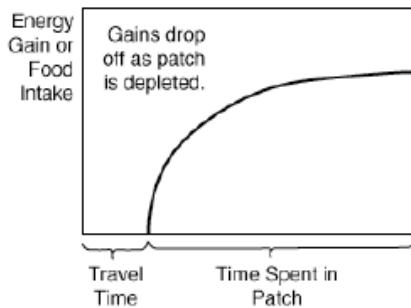


Figure 2.3: An energy gain function

The aim of this proposal is evaluate subjects' innate ability to reproduce similar behaviours while foraging for information rather than for food. To do this, the context of a gold digging strategy game has been chosen, in order to remove all proximal cues that might depend on the user's previous experience. In this environment, the user will have to perform the same kind of cost vs. benefit choices, however, he would not be able to adopt the strategies he consciously would. Following I discuss a paper by Pirolli & Card on Information Foraging which seems to support this intuition proposing experimental evidence and mathematical models. "

2.1.2 Information Foraging Theory

Pirolli & Card seem to start from the assumption that there are many similarities between the techniques adopted by animals foraging in the wild and the ones adopted by people in "Information Foraging". Because of the structure of today's society, in fact, it is not for food that we forage for. Food is usually readily available almost anywhere human settlements can be found, however, to purchase it, we need to engage in a "complex tributary of cultural tasks that engage our physical and social environments" (Pirolli & Card 1995) which demand us to develop numerous "information-based" strategies, in order to earn a living. Pirolli & Card begin by explaining that our adaptive success seems to be increasingly dependent on our mastery of techniques of information-gathering, sense-making, decision-making and problem-solving strategies. In turn, they claim, this leads us to implement strategies which are similar to the ones seen in foraging behaviours of animals in the wild. Here, we can observe patterns in food foraging behaviours

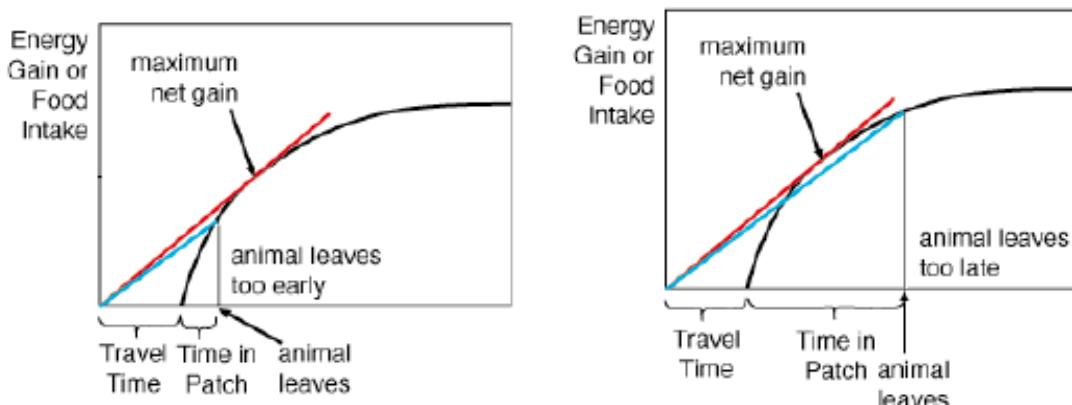


Figure 2.4: The optimal rate of gain

which aim at maximising the energy intake while minimising the amount of energy spent while foraging.

In other words, our lives are increasingly dependent on the ways in which we organise and retrieve information in order to use it effectively to navigate and survive today's social (and sometimes physical) landscape. Pirolli & Card go on to claim that, the Information Foraging theory's main tenant is that: "when feasible, natural information systems evolve towards stable states that maximise gains of valuable information per unit cost. Cognitive systems engaged in information foraging will exhibit such adaptive tendencies" (Pirolli & Card 1995).

As a consequence, it is to be expected that people will modify their information foraging strategies as well as the structure of the environment itself, whenever possible, to "maximise the rate of gaining valuable information" (Pirolli & Card 1995). Better strategies will be the ones that will allow someone who adopts them to yield more information per unit cost. Furthermore, it is also expected that these strategies will evolve through time, in order to reach a seemingly stable state that maximises the potential gains.

The process of analysing the development of this kind of behaviour, is called "adaptation analysis". This kind of analysis is conducted here, as well as in biology, through the use of "optimisation models" to study the design features of organisms and artefacts. Optimisation models include three major components:

- *Decision assumption*, determine how much time is to be spend analysing a certain collection of information as well as which kind of content is worthwhile pursuing.
- *Currency assumptions*, determine how the choices made through decision assumptions are to be evaluated. In the context of Information Foraging, the relevant currency will be amount of relevant documents found, as opposed to the amount of energy gained in food foraging strategies.
- *Constraint assumptions*, determine what kind of limits will apply to the relationship between decision and currency assumptions. In Information Foraging, these include (but are not limited to) previous knowledge, available technology and task structure.

These models allow us to construct a framework for the evaluation in Information Foraging, however, it is not to be expected that any single individual will fully be conscious and even evolve towards, an optimal awareness and implementation of these models. These models simply outline the possibility of "an advantageous adaptation if not blocked by other forces".

The main decision an organism has to make in its foraging endeavours (being it for food or for information) is determined by a problem of "Enrichment vs. Exploitation" of a certain "patch" of relevant documents (or food). The careful weighing of one against the other, will, in turn, determine its searching behaviour, comprehensive of "in-patch" and "between-patch" behaviours. When we decide to "enrich" a certain patch of information, we engage in certain "enrichment strategies" which are meant to maximise the amount of relevant information per unit of cost that we are able to get from a patch. However, the adoption of these strategies themselves has a cost which should be considered when making the decision to move to a different patch and select a new set of documents. There are two main "in-patch" enrichment strategies.

The first one is based on producing information packages that yield better results. This involves strategies like developing or acquiring better search tools, as well as spending time mastering them. The second one is based on filtering the incoming information into relevant topics or according to other decisional criteria devised by the foragers and that they came to realise as useful to their foraging needs.

Another way foragers can increase their gain in yielded by their foraging behaviours is by using "between-patch" enrichment strategies. These are also of two kinds. The first one is constituted by what Pirolli & Card call "scent-detection strategies". These strategies are based on the identification of proximal cues in the environment in order to make a choice on whether it would be fruitful to explore a certain "patch" or move to another one based on detection of proximal cues. In the context studied by Pirolli & Card, this translates into identifying the relevance of a certain document, or group of documents by, for instance, its title and payoff, perceived clarity of writing or length.

A second kind of in-patch enrichment strategy is based on reducing the cost of getting from one information patch to another. While this is usually impossible for animals in the wild, because it involves modifying the environment, it is instead a very efficient way for information foragers to improve the rate of gain per unit cost. In the example proposed by Pirolli & Card this is shown in the re-organisation of the workspace of an employee whose job is to write a Business Intelligence Newsletter. The subject of their experiment, in fact, organised the different areas of his office, in order to minimise the time spent searching for relevant document by disposing the material he knew he would need more frequently, the nearest to his working station.

For the purpose of this proposal it is also important to consider the conventional models of foraging on which Information Foraging Theory is based. As it can be expected, to provide an efficient model of foraging, we will have to take into account equations which model both in-patch and within-patch behaviours. Given what previously stated, Pirolli & Card start by characterising the rate of gain of valuable information per unit cost R as the ratio of the total net amount of valuable information gained G divided by the total amount of time spent between patches T_B and exploiting within patches T_W a

$$R = \frac{G}{T_B + T_W} \quad (2.1)$$

Notably, this equation assumes that (1) "the total amount of information gained can be represented as a linear function of between-patch foraging time:

$$G = \lambda T_B g \quad (2.2)$$

And that (2) the total amount of within-patch time can be represented as:

$$T_W = \lambda T_B t_W \quad (2.3)$$

This, in turn, gives Holling's Disc Equation:

$$\begin{aligned} R &= \frac{\lambda T_B g}{T_B + \lambda T_B t_W} \\ &= \frac{\lambda g}{1 + \lambda t_w} \end{aligned} \quad (2.4)$$

This formulation, however, "addresses the problem of allocation of time between in-patch vs. between patch under certain strong assumptions". Because of this, Pirolli & Card have to devise their own interpretation which takes in to account that "(a) there might be different kinds of patches and (b) the total gains from a patch depend on the within-patch foraging time which is under the control of the forager. For this reason they have to include a value i representing the type of patches encountered at a rate of λ_i and a value t_{wi} representing the *policy* adopted by a forager on how much time to spend within each patch. The total gain can then be represented as the sum of all the values of i between 1 and P.

$$\begin{aligned} G &= \sum_{i=1}^P \lambda_i T_B g_i(t_{wi}) \\ &= T_B \sum_{i=1}^P \lambda_i g_i(t_{wi}) \end{aligned} \quad (2.5)$$

In the same way, the total amount of time spent within patches could be represented as:

$$\begin{aligned} T_W &= \sum_{i=1}^P \lambda_i T_B t_{wi} \\ &= T_B \sum_{i=1}^P \lambda_i (t_{wi}) \end{aligned} \quad (2.6)$$

Combining these two equations according to (eq. 2.1) gives us:

$$\begin{aligned} R &= \frac{T_B \sum_{i=1}^P \lambda_i g_i(t_{wi})}{T_B + T_B \sum_{i=1}^P \lambda_i t_{wi}} \\ &= \frac{\sum_{i=1}^P \lambda_i g_i(t_{wi})}{1 + \sum_{i=1}^P \lambda_i t_{wi}} \end{aligned} \quad (2.7)$$

This last equation is what Pirolli & Card call the "patch model for information foraging" which takes into account patches that yield different kinds of gain functions as well as different strategies to exploit them.

Figures (a), (b) and (c) are the graphical representation of the functions presented in the previous section according to Charnov's Marginal Value Theorem, they model the problem of allocation of time between in-patch vs. between-patch strategies. Here, between-patch search times t_B , t_{B1} and t_{B2} generate different tangents to the $g(t_w)$ gain function which, in turn, determines the optimal rate of gain R . The outside of the curve on the x axis represents between-patch

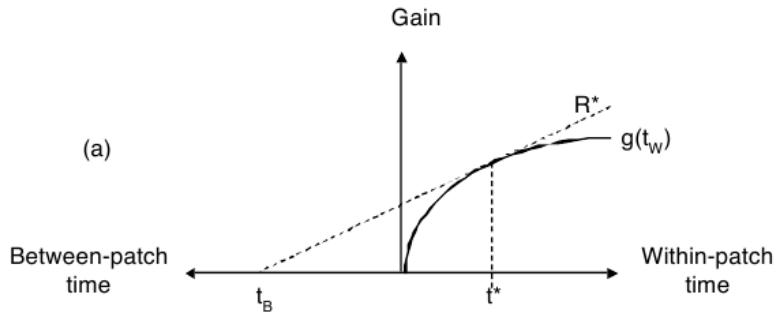


Figure 2.6: (a) - The Graph illustrates Charnov's Marginal Value Theorem where t^* denotes the optimal amount of time spent within-patch given the intersection between the gain function $g(t_w)$ and the tangent passing from t_B (the time spent between patches)

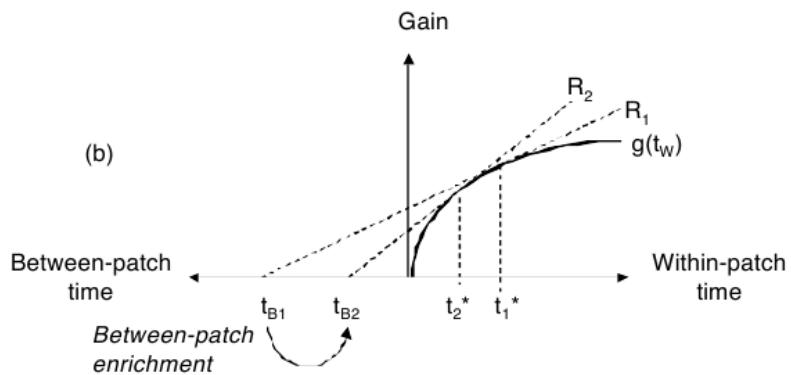


Figure 2.7: (b) - Shows the changes generated by the adoption of between-patch enrichment strategies. The tangent to the gain function $g(t_w)$ passing from t_{B2} determines that the optimal amount of time to spend in a given patch is now t_{2*} . The forager is now able to afford to spend less time in each patch.

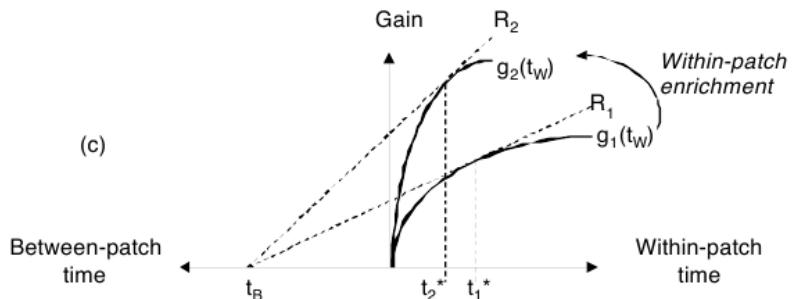


Figure 2.8: (c) - Shows the changes generated by the adoption of within-patch enrichment strategies. The new gain function $g_2(t_w)$ has a higher rate gain which allows foragers that spend the same amount of time between patches t_B will be able to reap more results in a shorter time.

time while, the inside of it, represent within-patch time. To determine the optimal rate of gain R_* one draws the tangent line to the gain function $g(t_w)$ and passing through t_B . The point of tangency will determine the "optimal allocation of within-patch foraging time t^* ".

This means that, the more time is spent between-patch, the less a forager will find it beneficial to spend time exploiting a given patch, always depending on the gain function $g(t_w)$. In Figure 2, in

fact, we see the effects of a between-patch enrichment like the one described previously (office re-disposition) which sets the optimal rate of gain in such a way that it will be more beneficial for the forager to spend more time within-patch to exploit it. Finally, in Figure 3 we can see the effect of within-patch enrichment (for instance, making information packages that yield better results). As we can see, within-patch enrichment, changes the gain function $g(tw)$ determining higher gains per unit of time spent within-patch.

The objective of the experiment outlined in this proposal, is to be able to evaluate the strategies adopted by subjects in order to reach an optimal rate of gain and to analyse how they are able to choose between within/between-patch enrichment strategies to better their rate of gain per unit cost.

2.2 Previous studies

2.2.1 Searching behaviour

2.2.2 Gamification

Chapter 3

Design

The design of Gold Digger went through different iterations through the course of its development (see 4.2 Heuristic Evaluation), however some design choices were made in order to keep the user experience consistent throughout the website and enhance clarity and ease of use.

Separation between "game pages" and "site pages"

Because Gold Digger is both a website and a game, it seemed appropriate to somehow separate the "game environment" from the more "site-like" features and pages, while still maintaining a certain degree of coherence between them. "Game pages" are the ones immediately relevant to the game in itself, the ones that the user will most likely enter while playing Gold Digger. These are: **the Game/Mine page, the General Shop page, the Home Page and the World Map Page**. The other pages are said to be "site pages" and they include pages like the About page and the Leaderboards, which a user is not very likely to access while in the middle of a play through. Furthermore "game pages" all require the user to be logged in to be accessed while the "site pages" do not.

Notwithstanding this distinction, the user experience is not disjointed thanks to prominent recurring elements such as the nav bar on the top of the page and the landscape headline showing the name of each page.

3.1 Site Map

Navigation through the website is aided by a navbar located at the top of every page (with the exception of the 'game over' and 'end of the day' pages).

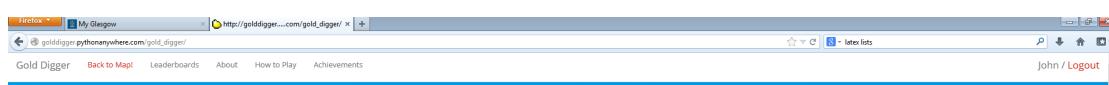


Figure 3.1: Gold Digger nav bar

From the navbar the user will be able to reach the following pages:

- The main page (by clicking on 'Gold Digger')
- The World Map page (if signed in)

- The Leadeboards
- The About page
- The 'How to Play' / Tutorial page
- The 'Achievements' page
- The User Profile page (if signed in)

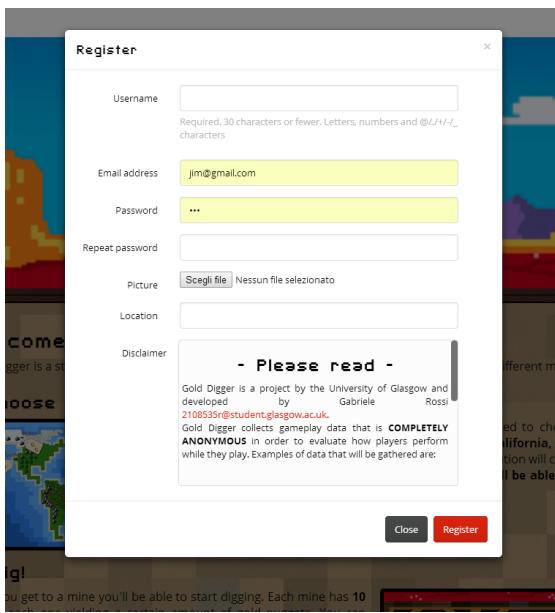
finally, the user will also be able to logout at any moment by using the 'Logout' button on the top right. However, if the user is currently in one of the mines, clicking on one of these links will result on a warning message being displayed that will alert users that the gold gathered during the day will be lost if they leave the mine before exhausting the time at their disposal and reaching the end of the day.

3.1.1 Home, Registration and Login



Figure 3.2: Home Page

On the landing page (homepage) users will be presented with a short **five-point explanation** of the game to quickly explain the mechanics of the game so that users could start playing as soon as possible. To do this, they will have to login through the form on the left or register by clicking on the 'Register' button.



Clicking on the '**Register**' button will trigger a modal asking users to create a new user-name and password, as well as optionally entering their location and user picture. Finally, the modal displays a disclaimer in order to both make sure that the users are aware of the limitations of accessing the website and its purpose. Users who wish to have more information about Gold Digger are redirected to the 'About' page or offered a link to directly write an email to the developer.

Finally if a user enters wrong details or forgets to fill in a required field, (both for registration and login) an appropriate error message is clearly displayed for the user to see and try again. At present there is no limit to the number of attempts a user can make at logging in or registering.

Figure 3.3: Registration modal

3.1.2 World Map

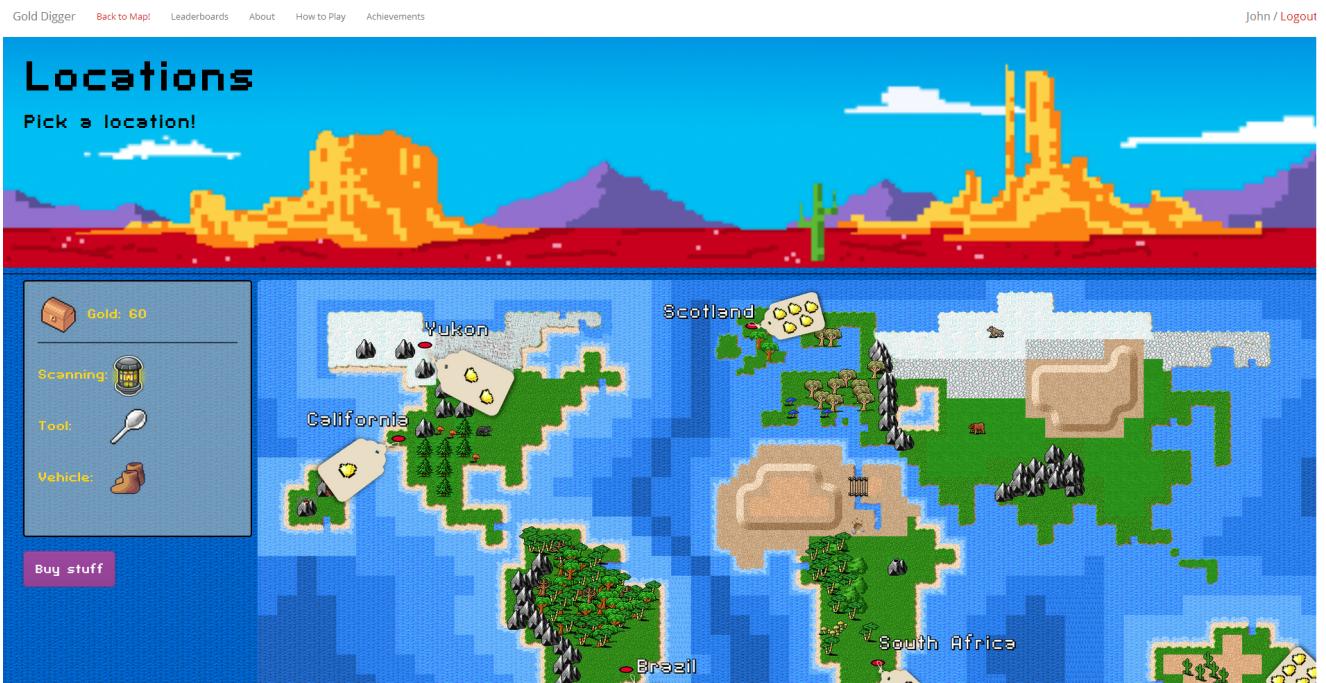
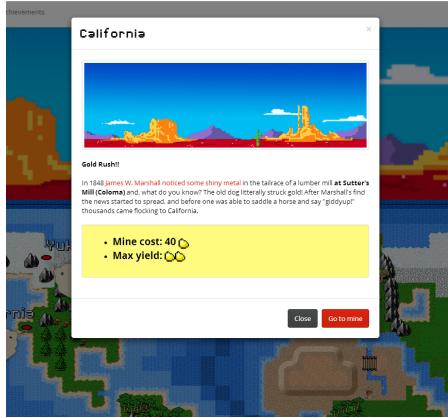


Figure 3.4: World Map

Once users have registered or have been logged in, they are automatically redirected to the world map so that they can start playing immediately. On the left of the page, users can see the euqipment and the amount of gold they have at the moment. The side item panel appears on the left column of the page, once the user has logged in. The panel shows the items that users have equipped, together with the total amount of gold in their possession. By hovering

over each one of the items a tooltip will appear, showing the essential stats of each item. The presence of the side item panel has the function of both confirming the users that they are logged in and reminding them that their game session has started contributing to the uniformity of the user experience.



On the right hand side of the page users can access any of six locations: **California, Yukon, Brazil, South Africa, Scotland** and **Victoria**. Clicking on any of the locations will trigger a modal that displaying the scenery of the particular mine, some information about the gold digging history in that region (with links to Wikipedia articles), the cost of the mine and the amount of gold the user can expect to find. Starting from California and ending with Victoria, each mine is more expensive than the previous one but it also have a potentially higher gold yield. It is up to the player to enter the right mine at the right time.

Figure 3.5: California modal

3.1.3 Game Screen

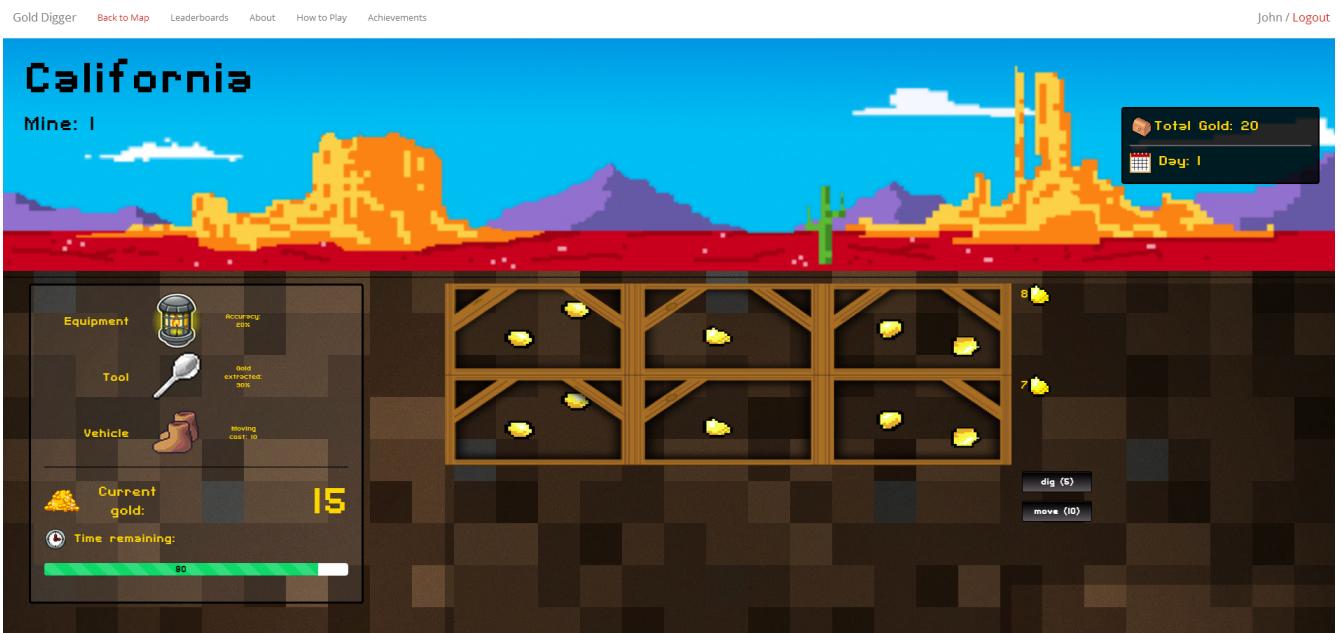


Figure 3.6: Mine

Once the users enter a mine, the appropriate amount of gold is removed from the total and they can start digging. Each mine presents a similar structure, except for the landscape and the amount of gold that can be found in each layer. On the top of the page we can see the landscape picture and the name of the location, as well as the number of mines that the user has dug into during the present day. As previously noted, on the left, we can find the equipment panel, in the middle we have the mine shaft and on the right have the '**Dig**' and '**Move**' as well as the yield of each layer that has been already dug. The left side panel contains here some more information than it does in the other pages where it appears:

- **Current gold:** the amount of gold gathered during the present day.

- **Time remaining:** the amount of time (in units of time) remaining before the end of the day.

Furthermore, because the left side panel's position is fixed, it will follow users as they get deeper and deeper in the mine, allowing them to always keep an eye on the game state without having to go back to the top of the page to check how many units of time they have left before the end of the day. Because the 'Current gold' is summed to the 'Total Gold' only at the end of the day, the total amount of gold, together with the number of days users have been digging without encountering a game over, is displayed in a small box on the top right of the page. The options available to the user and an example of gameplay are detailed in section 3.2.

3.1.4 General Store

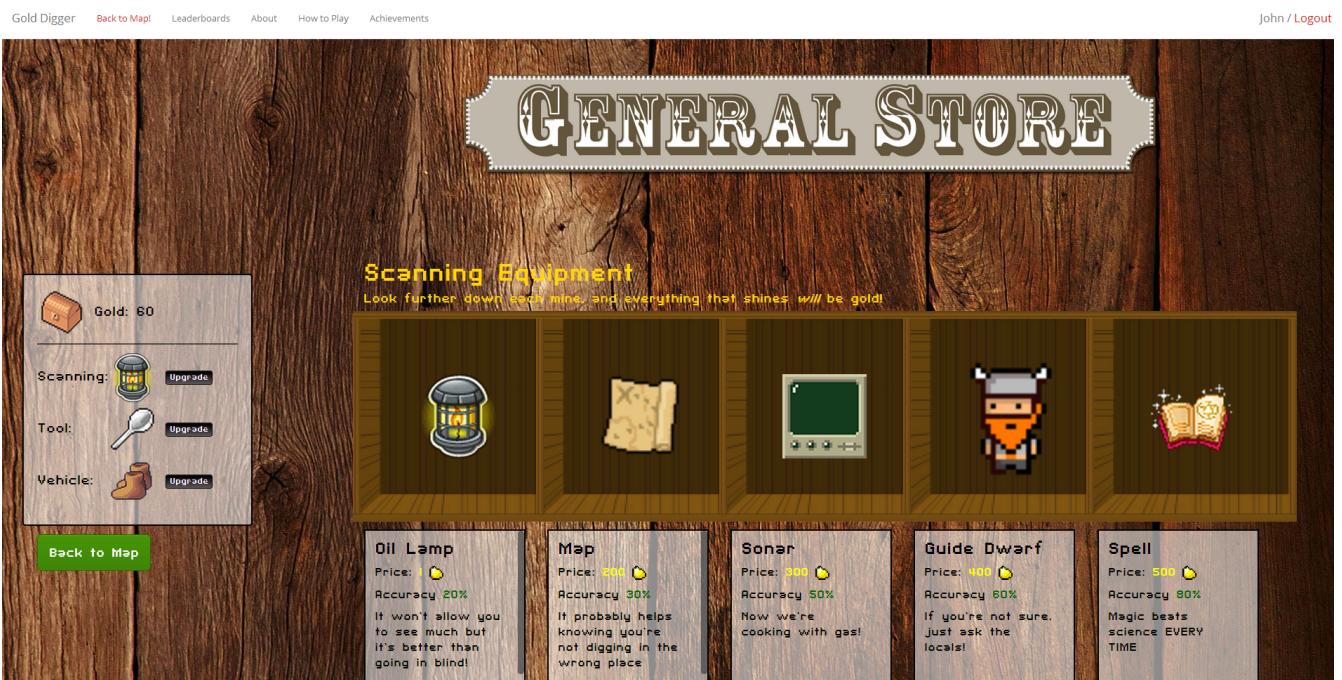


Figure 3.7: General Store

At the end of each day (by clicking on '**shop**'), or from the world map screen (by clicking on '**Buy stuff**'), users can access the **General Store** (see fig. 3.7). From here they are able to browse and purchase new items to help them in their digging. There are three groups of objects that the user can choose from **Scanning Equipment**, **Digging Equipment** and **Vehicles**. Each of these groups contains five different items with different cost and stats. Users are able to see the image associated with each item as well as its cost, stats and a short description in the small tex box underneath. If users decide to upgrade one of their item groups (Scanning, Digging or Vehicle) they can do so by clicking on the '**Upgrade**' button. At this point they will be presented with a small modal that reviews the item's stats, together with its picture, asking the user to confirm the purchase. If the users don't have enough gold to make the purchase, or if the purchase would cause them to not have enough money to enter the cheapest mine, an appropriate alert message is displayed, explaining why the purchase is not possible.

The choice to allow users to *upgrade* rather than *buy* items has been made because each of the items is better than the previous one in every respect and thus there is no reason why a user would chose to equip an item that they previously purchased, since this would not bring

any advantage at all. For instance, with digging tools, their cost, dig cost (the amount the user has to pay in time units to dig once) and extraction power (the percentage of gold that the user will be able to extract given a certain yield), are all increased from the least, to the most expensive. However, it would be easy to add items that have different combinations of these parameters and add, for example, a digging tool that, at a higher cost per dig also returns a higher percentage of gold. Finally, upon purchase, the new item is immediately added to the left side panel and the appropriate amount of money removed from the user's total through an AJAX call.

3.1.5 Leaderboards

Average	Max Gold	Days Worked	All Time Gold	Achievements	Average Gold Per Mine	All Time Gold Dug	Max Gold Reached	Days Dug
Rank	Thumbnail	Username	Equipment					
1		Gareyden			272.0	83602	56060	60
2		koocharrs			271.0	97392	71113	77
3		Tina			197.0	32187	9964	56
4		s.andi			163.0	16135	3622	32
5		luke			159.0	14532	3030	28

Figure 3.8: Leaderboards

On the **Leaderboards** (see fig. 3.8) page, users are able to check their performance and compare it to the other players'. There are four parameters by which users can be ranked, plus an achievements board that has no specific ranking criteria:

- **Average:** the total amount of gold ever dug by the player divided by the total amount of mines that were dug into.
- **Max Gold:** the maximum amount of Gold ever dug (the maximum amount of gold ever reached)
- **Days Worked:** the total amount of days of digging completed by the player (not zeroed on game over)
- **All Time Gold:** the cumulative total amount of gold dug by the player (not zeroed on game over)
- **Achievements:** shows the achievements gained by each player in no particular order (achievements are not ranked)

Each row of the the tables (excluding the 'Achievement' table) diplays the following parameters: Rank, Thumbnail, Username, Equipment (showing the image for each of the three item groups), Average Gold per Mine, All Time Gold Dug, Max Gold Reached, Days Dug.

3.1.6 Tutorial

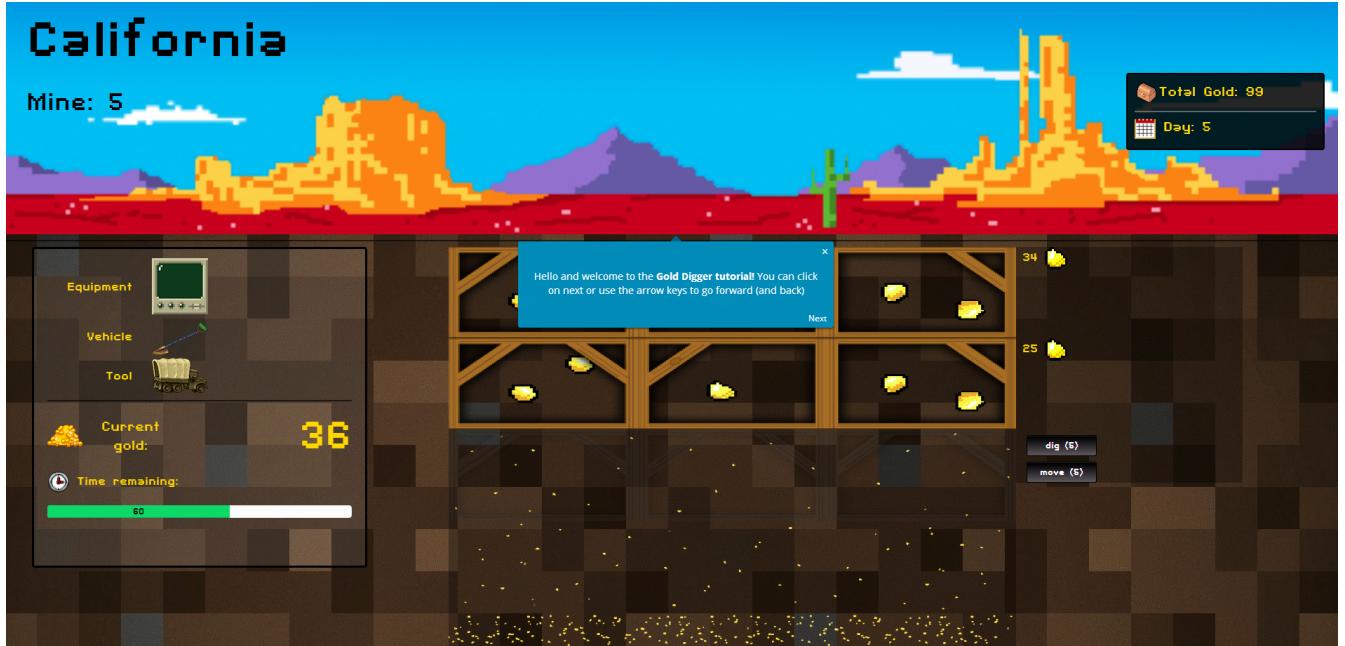


Figure 3.9: Tutorial

Because it is important for users to have a clear idea of the way the game works, all effort has been made to employ a visual approach that would be quick and easy to understand. To this end, upon entering, the '**How to Play**' page, a tour of the main game features is automatically launched. The javascript library **Trip.js** (see 4.4.6) highlights and points at the different parts of the game screen in order to get a quick visual explanation of the main game mechanics. Each of the labels has 'Next', and 'Previous' link, in order to let users go back to a previous point or skip explanations they already viewed.

3.1.7 Achievements

The '**Achievements**' page dispays all the achievemts badges together with their name, the condition that triggers them and their image. This page is needed in order for the users to be able to check which achievements they can aim for while playing. Two additional achievements (fig. 3.11) are not displayed here and are given only under special conditions:

- **Awesomeness**: given to the people who tested the "beta" version of the site.
- **Banana**: given to people who managed to find the game's Easter Egg

Achievements ans special achievements have been incorporated in order to both add an extra game feature to Gold Digger which wouldn't change the game mechanics too much an to give players an incentive to keep playing, especially since there is no particular "winning condition", fulfilled which a player has completed the game.

3.1.8 About Page

In this page, users are able to find the developer and supervisor's contact details as well as reading about the reasons behind the creation of Gold Digger and acknowledgements of the authors of some of the material used in the site.

3.2 Walkthrough

(1) Jill, the user, has just logged in (or registered) and has been redirected to the World Map. Her starting Equipment and Gold are as follows:

- **Gold:** 100
- **Scanning Tool:** Oil Lamp (accuracy: 20%, visibility: 2)
- **Digging Tool:** Spoon (dig cost: 5, gold extracted 30%)
- **Vehicle:** Boots (move cost: 10)

On the map, Jill clicks on California and a modal appears, from which she can see that, although the mine doesn't yield much gold, it is relatively cheap to access it (she only needs to pay 40 gold nuggets to enter it).

(2) After clicking on 'Go to Mine', 40 gold nuggets are removed from Jill's total amount of gold and she is presented with a mine where no layer has been dug. On the top left hand side Jill can see the name of the location she is in and the number of the mine (now 1) whereas on the right hand side she can check her total amount of gold (now 60) and the number of days she

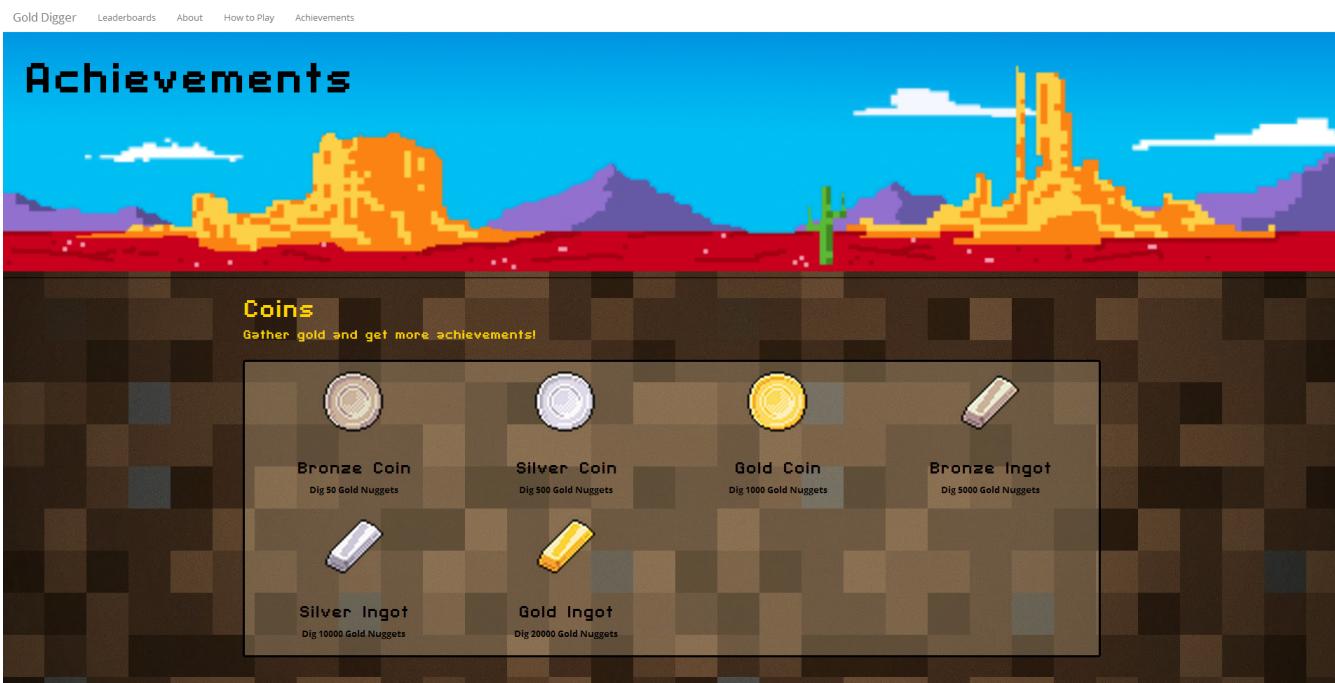
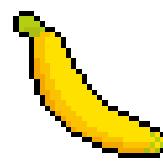


Figure 3.10: Achievements display

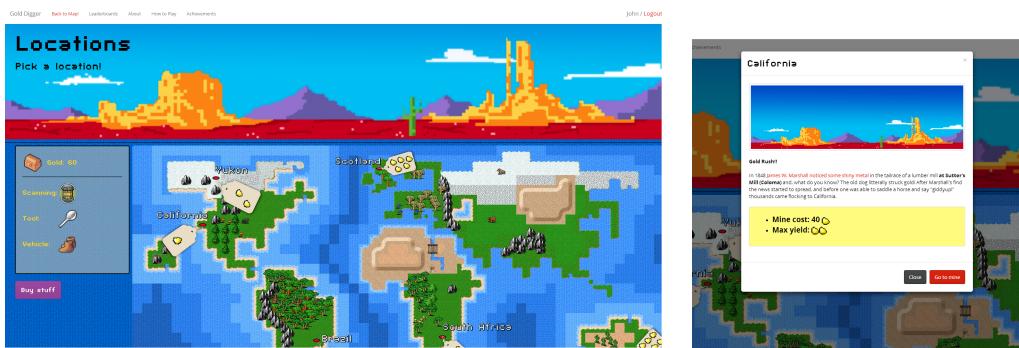


(a) You helped testing!



(b) You found the Easter Egg!

Figure 3.11: Special Achievements



has been digging. Thanks to her Oil Lamp she can see specks of gold in the first two layers but, because the lamp only has a 20% accuracy, it is possible that the amount of flecks that can be seen is deceiving.

- **Current gold:** 0
- **Time remainingl:** 100
- **Mine:** 1
- **Day:** 1

(3) Jill decides to dig through the first couple of layers layer of the mine by clicking on the 'Move' button on the right hand side of the mine shaft. She gains 17 gold nuggets and consumes 10 units of time (since each digging operation costs 5 units of time)

- **Current gold:** 17
- **Time remainingl:** 90
- **Mine:** 1
- **Day:** 1

(4) At this point Jill is not sure she might get much more gold from this mine and she doesn't want to spend precious time digging through layers of the mine that she has no information about. For this reason she clicks on 'Move' to get to a new mine (in the same location). To do this she must spend 10 units of time

- **Current gold:** 17
- **Time remainingl:** 80

- **Mine:** 2
- **Day:** 1



(5) Jill continues to dig through some more mines and after a while she runs out of units of time. At this point, she is presented with the 'End of the Day' screen that sums up her progress during the day as follows:

- **Today's gold:** 94
- **Total gold:** 94
- **Mines dug:** 6

Jill also receives the 'Bronze Coin' achievement for having dug more than 50 gold nuggets. From here Jill decides to go to the shop and see if she can purchase some new items



(6) Once in the shop Jill tries to purchase better scanning equipment, so she clicks on the 'Upgrade' button next to the Oil Lamp. Unfortunately she doesn't have enough gold to make this purchase yet, so a message alerts her that she would need to do some more mining if she wants to be able to purchase the item.

(7) To gather more gold, Jill goes back to the California mines and, as soon as she has 100 gold nuggets, she takes a gamble and spends them on accessing the Yukon mine. However, she doesn't perform very well and, at the end of the day she owns less than 40 gold nuggets. Because she doesn't have enough money to enter any of the mine, Jill loses her first game. However, clicking on 'Back to Map' will restore her to the initial conditions, if she bought any items they wold be lost.



3.3 System Architecture

Gold Digger is based on a 3-tier architecture. Each user is able to create a player account (client) containing all of the details entered upon registration, together with a set of game variables, set to their starting values. The client side renders HTML5 and CSS3 elements and graphics while updates to the game states are done mostly through AJAX, in order to avoid reloading the page. The game logic is coded in the middleware using the python-based Django framework (see 4.4.1). All of the user profiles as well as the other game objects and achievements are stored in the database while the corresponding graphics, as well as the rest of the game graphics are stored in the project's static and media folders. Finally, the middleware continuously updates a log file in order to monitor user behaviour and gather data to be parsed and analysed at a later stage.

3.4 ER Model

The above diagram describes the entity-relationship model for Gold Digger. There are 5 entities in Gold Digger:

- **User Profiles:** this entity contains several of the users' game stats, like the amount of gold at their disposal and the average amount of gold dug per mine as well as a reference

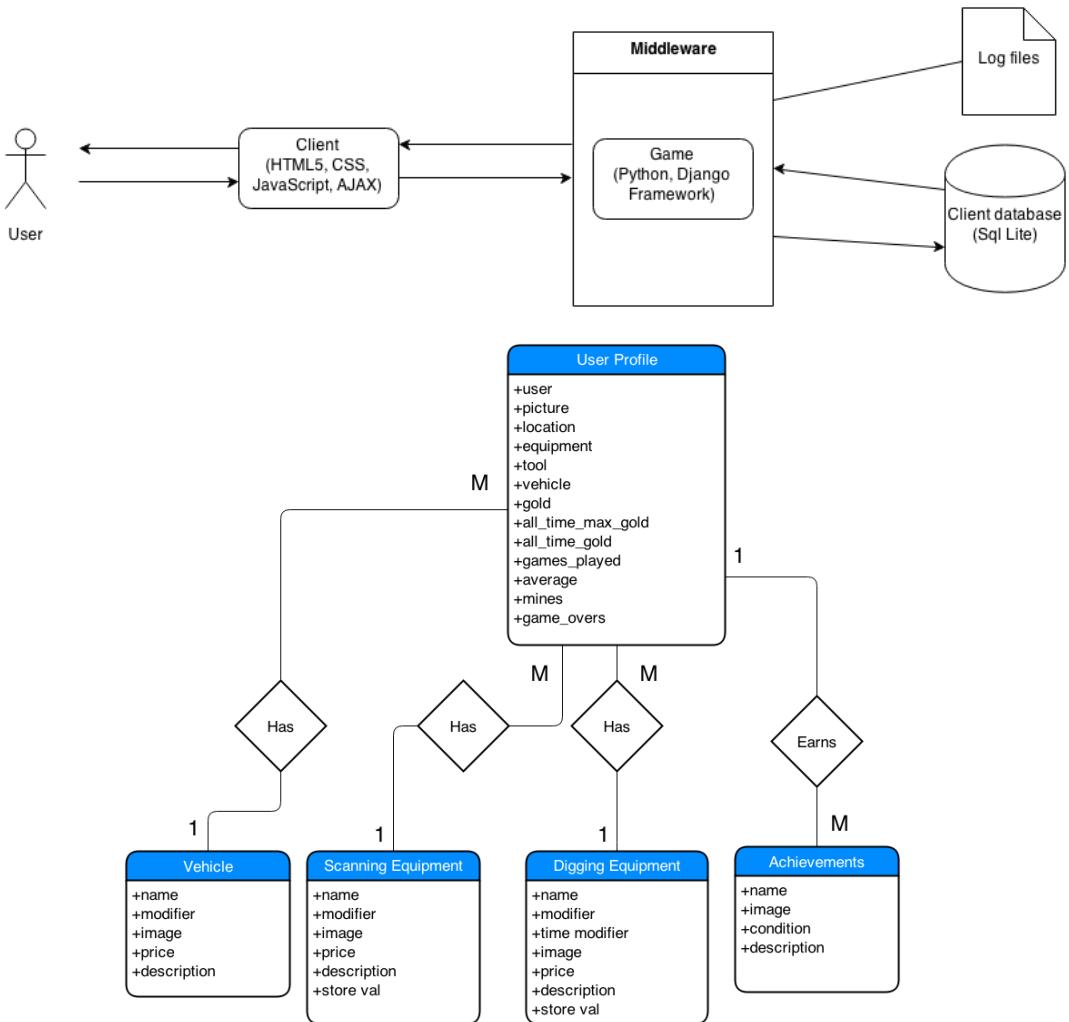


Figure 3.16: The Gold Digger ER diagram

to the Django object 'user' containing all the information regarding the user's password, username and email. This is necessary in order to take advantage of Django's very efficient user management system and keep the website as safe as possible

- **Digging Equipment:** this entity contains each of the tools' name, image, price and description. However, the most important attributes of this entity are the *modifier* and the *time modifier*. The *modifier* attribute is used by the game mechanics to calculate how much gold a user is able to extract from a given layer of the mine. It is a *float* number between 0.2 and 0.8 that will be multiplied by each of the amounts of gold in each layer. This way if the user is using, for instance, a shovel, she will be able to extract only 30% of the gold that is actually present in that layer (see ??). The *time modifier* attribute, on the other hand stores the digging cost per digging operation of each of the tools and it is an integer that goes from 5 to 1 in decreasing order from the worst tool to the best.
- **Scanning Equipment:** this entity is quite similar to Digging Equipment but it does not include the *time modifier* attribute. The *modifier* attribute of this entity is responsible for calculating the cue array, that is in turn responsible for determining the amount of gold flecks that will be shown in each of the mine's layers (see ??). Furthermore, because this attribute holds only float values from 0.2 to 0.8, a simple multiplication by 10 is used to determine the number of layers in which those specs are actually visible .

- **Vehicles:** this entity, like Scanning Equipment, only possesses the *modifier* attribute. This is an integer in range 10 to 6 and it holds the value of time units to be subtracted from the day's time, for each digging operation.
- **Achievements:** the Achievement entity simply stores the name, image, condition and description of each of the achievements.

Relations

There are **four relations** between the above entities. Each player can only store one relation to a particular Scanning Equipment, Digging Equipment and Vehicle object thus, these are all one-to-one relationships. However, because each user can earn many achievements, the relationship between the User Profile and the Achievement entity needs a separate database table to keep track of the user-achievement value couples. In this table, **User Achievements**, each row comprises two columns, each of which is a foreign key. The first one is the relation to a User Profile Object, and the second one is the relation to an Achievement object.

3.5 Graphics

One of the main elements of the design of Gold Digger was its graphic design. The general idea was to reproduce the look and feel of a retro 8-bit game as much as possible, without compromising the clarity and ease of use needed by a website. For this reason, 8-bit-style graphics were employed in most of the elements of the 'game pages' while Twitter Bootstrap was used in order to provide a simple minimalistic "framework" for the graphics to be displayed in.

Fonts

There are two fonts that have been used throughout Gold Digger. The first one is **Open Sans** (see 4.4.5). This font is part of the Twitter Bootstrap theme that provides the graphic framework for the site. It is used in areas with lots of text and in the navbar, where clarity is the most important issue. However, throughout the site, most of the headings, scores, descriptions and game messages employ the font **SF Pixelate** (by ShyFonts) which has a pixelated look reminiscent of retro 8-bit games. All of the text in the 'game pages' is typed using this font.

Landscapes and World Map

There are six different landscapes in Gold Digger, each one associated with one of the locations that the user can travel to and dig into. Three of them (California, Scotland and Yukon) were created with a pixel-art specific (Pixelen) program using a number of pictures for inspiration, while the remaining three have been produced by applying filters, modifying and coloring similar pre-existing pictures in Photoshop CS, to achieve the 'pixelated look'.

The World Map was created by importing a set of freeware game tiles into a freeware level design tool called 'Tiled' (see A). This tool allows users to load a particular set of tiles and arrange them on a grid of their choosing. The tileset used for the World Map was originally created for the expansion and creation of new maps for the game "Tales of Middle Earth" but it lent itself very well to the task of creating a world map. After the first version of the map was completed, the location names and price tags were added with Photoshop CS 6 where other minor adjustments were also made.

Items and Achievements

Most of the items and achievements icons were made by Alis, a pixel artist who made them available for public usage (see A). However a minor percentage of the items were created by modifying sprites of the popular game METAL SLUG by SNK/Playmore. These sprites are all properties of their respective owners and no copyright is claimed upon them.

Chapter 4

Implementation

4.1 Development Methods

4.2 Heuristic Evaluation

4.3 Testing

4.3.1 Unit Testing

4.3.2 Live Testing

4.4 Technologies

4.4.1 Django

Django is a very powerful, well documented, easy to use, extensive web development framework based on python. In Gold Digger, it provides the base on which all of the other technologies employed are built upon. The choice of Django, as a web development framework has been made for all the reasons listed above, but some were important in particular. Firstly, Django uses a Model View Template architecture. Each url request is handled by the Model middleware in Django and redirected to a specific View associated with it. The view, in turn, handles the request and provides the data to be passed to a the Template that further processes it and displays it (see below for Django template code).

Another of Django's main features is the handling of SQLite databases through simple python commands. Because each user is associated with a User Profile object, frequent update of the database was needed in order to update all the game values and stats. The code snippet below, for example, is responsible for assigning an achievement to a player. This requires the method to be passed a reference of the user as well as reference to the achievement she gained and then, if the user hasn't earn the achievement already, create a new row in the User Achievement

database table (see 3.4. Django allows us to do this very easily, by simply checking if the user-achievement pair already exist in the database (on line 4) and, if not, go ahead and create it. As we can see, this only takes a few lines of very simple code. In the code snippet, the method returns a object (*myResponse*) containing all the information regarding the achievement in order to be sent back as a response to an AJAX call (see 4.4.7) and be subsequently displayed to the user.

```

1 myResponse = {}
2 if not UserAchievements.objects.filter(user=user, achievement=achievement).exists():
3     achieve = UserAchievements()
4     achieve.user = user
5     achieve.achievement = achievement
6     achieve.save()
7     print "Achievement_UNLOCKED"
8
9
10    myResponse[ 'achievement_name' ] = achievement.name
11    myResponse[ 'achievement_condition' ] = achievement.condition
12    myResponse[ 'achievement_image' ] = achievement.image.url
13    myResponse[ 'achievement_desc' ] = achievement.description
14
15    return myResponse
16 else:
17     myResponse[ 'unlocked' ] = True
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
188
189
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
209
210
211
212
213
214
215
216
217
218
219
219
220
221
222
223
224
225
226
227
228
229
229
230
231
232
233
234
235
236
237
237
238
239
239
240
241
242
243
244
245
246
247
247
248
249
249
250
251
252
253
254
255
256
256
257
258
259
259
260
261
262
263
264
265
266
267
267
268
269
269
270
271
272
273
274
275
276
277
277
278
279
279
280
281
282
283
284
285
286
287
287
288
289
289
290
291
292
293
294
295
296
297
297
298
299
299
300
301
302
303
304
305
306
307
307
308
309
309
310
311
312
313
314
315
315
316
317
317
318
319
319
320
321
322
323
324
325
325
326
327
327
328
329
329
330
331
332
333
334
335
335
336
337
337
338
339
339
340
341
342
343
344
344
345
346
346
347
348
348
349
349
350
351
352
353
353
354
355
355
356
357
357
358
359
359
360
361
361
362
363
363
364
365
365
366
367
367
368
369
369
370
371
371
372
373
373
374
375
375
376
377
377
378
379
379
380
381
381
382
383
383
384
385
385
386
387
387
388
389
389
390
391
391
392
393
393
394
395
395
396
397
397
398
399
399
400
401
401
402
403
403
404
405
405
406
407
407
408
409
409
410
411
411
412
413
413
414
415
415
416
417
417
418
419
419
420
421
421
422
423
423
424
425
425
426
427
427
428
429
429
430
431
431
432
433
433
434
435
435
436
437
437
438
439
439
440
441
441
442
443
443
444
445
445
446
447
447
448
449
449
450
451
451
452
453
453
454
455
455
456
457
457
458
459
459
460
461
461
462
463
463
464
465
465
466
467
467
468
469
469
470
471
471
472
473
473
474
475
475
476
477
477
478
479
479
480
481
481
482
483
483
484
485
485
486
487
487
488
489
489
490
491
491
492
493
493
494
495
495
496
497
497
498
499
499
500
501
501
502
503
503
504
505
505
506
507
507
508
509
509
510
511
511
512
513
513
514
515
515
516
517
517
518
519
519
520
521
521
522
523
523
524
525
525
526
527
527
528
529
529
530
531
531
532
533
533
534
535
535
536
537
537
538
539
539
540
541
541
542
543
543
544
545
545
546
547
547
548
549
549
550
551
551
552
553
553
554
555
555
556
557
557
558
559
559
560
561
561
562
563
563
564
565
565
566
567
567
568
569
569
570
571
571
572
573
573
574
575
575
576
577
577
578
579
579
580
581
581
582
583
583
584
585
585
586
587
587
588
589
589
590
591
591
592
593
593
594
595
595
596
597
597
598
599
599
600
601
601
602
603
603
604
605
605
606
607
607
608
609
609
610
611
611
612
613
613
614
615
615
616
617
617
618
619
619
620
621
621
622
623
623
624
625
625
626
627
627
628
629
629
630
631
631
632
633
633
634
635
635
636
637
637
638
639
639
640
641
641
642
643
643
644
645
645
646
647
647
648
649
649
650
651
651
652
653
653
654
655
655
656
657
657
658
659
659
660
661
661
662
663
663
664
665
665
666
667
667
668
669
669
670
671
671
672
673
673
674
675
675
676
677
677
678
679
679
680
681
681
682
683
683
684
685
685
686
687
687
688
689
689
690
691
691
692
693
693
694
695
695
696
697
697
698
699
699
700
701
701
702
703
703
704
705
705
706
707
707
708
709
709
710
711
711
712
713
713
714
715
715
716
717
717
718
719
719
720
721
721
722
723
723
724
725
725
726
727
727
728
729
729
730
731
731
732
733
733
734
735
735
736
737
737
738
739
739
740
741
741
742
743
743
744
745
745
746
747
747
748
749
749
750
751
751
752
753
753
754
755
755
756
757
757
758
759
759
760
761
761
762
763
763
764
765
765
766
767
767
768
769
769
770
771
771
772
773
773
774
775
775
776
777
777
778
779
779
780
781
781
782
783
783
784
785
785
786
787
787
788
789
789
790
791
791
792
793
793
794
795
795
796
797
797
798
799
799
800
801
801
802
803
803
804
805
805
806
807
807
808
809
809
810
811
811
812
813
813
814
815
815
816
817
817
818
819
819
820
821
821
822
823
823
824
825
825
826
827
827
828
829
829
830
831
831
832
833
833
834
835
835
836
837
837
838
839
839
840
841
841
842
843
843
844
845
845
846
847
847
848
849
849
850
851
851
852
853
853
854
855
855
856
857
857
858
859
859
860
861
861
862
863
863
864
865
865
866
867
867
868
869
869
870
871
871
872
873
873
874
875
875
876
877
877
878
879
879
880
881
881
882
883
883
884
885
885
886
887
887
888
889
889
890
891
891
892
893
893
894
895
895
896
897
897
898
899
899
900
901
901
902
903
903
904
905
905
906
907
907
908
909
909
910
911
911
912
913
913
914
915
915
916
917
917
918
919
919
920
921
921
922
923
923
924
925
925
926
927
927
928
929
929
930
931
931
932
933
933
934
935
935
936
937
937
938
939
939
940
941
941
942
943
943
944
945
945
946
947
947
948
949
949
950
951
951
952
953
953
954
955
955
956
957
957
958
959
959
960
961
961
962
963
963
964
965
965
966
967
967
968
969
969
970
971
971
972
973
973
974
975
975
976
977
977
978
979
979
980
981
981
982
983
983
984
985
985
986
987
987
988
989
989
990
991
991
992
993
993
994
995
995
996
997
997
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
155
```

Finally, because some variables associated with the user are stored in the Gold Digger database as discussed above, however, some other variable will need to change very often during the game but still need to be associated with each particular user. These are variables such as the user's current gold, the array of layers of each mine, and the number mines the user has been in. For them, *sessions* need to be implemented. Thankfully Django also offers session support:

```
1 request.session[ 'has_mine' ] = False
2 request.session[ 'mine_type' ] = ''
3 request.session[ 'time_remaining' ] = 100
4 request.session[ 'gold' ] = 0
```

The code snippet above is employed in the view that handles the choice of a location in the world map. Here the session values *hasmine* and *mine_type* are set to 'False' and 'null' so that a new mine can be created by the game view, the time remaining is set to 100 (beginning of the day) and the current gold is set to 0, since the player hasn't started digging in this particular location yet. Using Django, these variables are associated with the particular user session and can be used by all the other view without being singularly passed.

4.4.2 Python

4.4.3 HTML5

4.4.4 CSS3

Animate.css

4.4.5 Twitter Bootstrap

4.4.6 Javascript and JQuery

Trip.js

Animatenumbers.js

4.4.7 AJAX

4.4.8 Code

Chapter 5

Results and Data

5.1 Data Logged

5.2 Data Analysis

5.3 Results

5.4 Reflection

Appendix A

First appendix

A.1 Section of first appendix

Appendix B

Second appendix

Bibliography