

UNIVERSITY OF „ALEXANDRU IOAN CUZA” IAŞI

Faculty of Computer Science



BACHELOR'S THESIS

# Survive

proposed by

*Ioana Parfene*

**Session:** *july, 2021*

Coordinator

Lect. Dr. Alex Mihai Moruz

UNIVERSITY OF „ALEXANDRU IOAN CUZA” IAŞI

Faculty of Computer Science

# Survive

*Ioana Parfene*

**Session:** *july, 2021*

Coordinator

Lect. Dr. Alex Mihai Moruz

# Contents

<b>Introduction</b>	<b>3</b>
Motivation . . . . .	3
Gameplay Description . . . . .	3
Requirements . . . . .	3
Approach . . . . .	4
Related Work . . . . .	4
<b>Contribution</b>	<b>6</b>
Time Frame . . . . .	6
Design Stages . . . . .	6
Technologies . . . . .	6
<b>1 Game Design</b>	<b>8</b>
1.1 Methodology . . . . .	8
1.2 User Interface . . . . .	9
1.2.1 Screen Navigation Architecture . . . . .	9
1.2.2 Screens . . . . .	9
<b>2 Gameplay</b>	<b>13</b>
2.1 Objective . . . . .	13
2.1.1 Environment . . . . .	13
2.1.2 Time . . . . .	13
2.1.3 Weather . . . . .	13
2.1.4 Day Time . . . . .	14
2.1.5 Shelter . . . . .	14
2.1.6 Locations . . . . .	14
2.2 Status Bars . . . . .	14
2.2.1 Condition . . . . .	15
2.2.2 Body Heat . . . . .	15
2.2.3 Hydration . . . . .	16
2.2.4 Calories . . . . .	16
2.3 Inventory . . . . .	16
2.3.1 Items . . . . .	17

2.3.2	Spoilage . . . . .	18
2.4	Fire . . . . .	19
2.4.1	Fire Starting . . . . .	19
2.5	Crafting . . . . .	20
2.6	Location Specific Actions . . . . .	21
2.6.1	Exploring . . . . .	21
2.6.2	Getting Wood . . . . .	22
2.6.3	Getting Water . . . . .	22
2.6.4	Setting Traps and Catching Fish . . . . .	23
2.7	Travel . . . . .	23
<b>3</b>	<b>Implementation</b>	<b>24</b>
3.1	Technologies . . . . .	24
3.1.1	Kivy . . . . .	24
3.1.2	Pickle . . . . .	24
3.1.3	Buildozer . . . . .	24
3.2	Integration . . . . .	25
3.2.1	Kivy Using Kivy Files . . . . .	25
3.2.2	Kivy Using Python Files . . . . .	26
3.3	Testing . . . . .	27
3.3.1	Quality Assurance . . . . .	27
3.3.2	Playtesting . . . . .	27
<b>Conclusion</b>		<b>28</b>

# Introduction

**Survive** is a take on the game **Survive - Wilderness survival**[1] by **Juuso Hietalahti**[2].

## Motivation

My favourite **programming direction** discovered in University has been **game development**. As an avid **lifelong player**, I have always brushed away the idea of **pursuing** it as more than a **hobby**. The practice is still quite **uncommon** in the **courses** and **career paths** found in Romania.

Throughout **college**, I managed to obtain some sort of **game system assignment** for the majority of the **mandatory projects** I had. The **Game Design**[3] course created by my **coordinator** was the reason I finally decided to take it seriously and choose a **game** for my **Bachelor's Thesis**.

My favourite games belong to the **survival**, **strategy** and **simulation** genres. I have been playing **Survive - Wilderness survival** for many years, even though I enjoy **PC games** much more. My **interest** and its **complexity** made it an **attainable, good choice** for my **Thesis**.

## Gameplay Description

**Survive** a long, challenging, resource-scarce **journey** back to the main road after getting lost in the woods on a rainy day. **Forage** for wood, berries, water and vines. **Hunt, fish, rest, cook** and **craft** campfires, weapons, traps, bandages and clothing insulators.

**Travel**, take different turns and make **strategic** choices in order to reach civilization before it is too late. Balance **hunger, temperature, thirst, health** and inventory to avoid **starvation, hypothermia, dehydration and illness**.

Stay alive or **permanently lose** the game and start over!

## Requirements

The **project** must be an **Android** application with an interface that recognizes **one finger screen touch**. It should be easy to **download** and to **install** on a **mobile device**.

The **player** should be allowed to **pause, resume, save** and **quit** the game quickly. **Exiting** has to lead to **discarding** unsaved **progress** and **dying** must result in a **game**

over, as one of the main design characteristics is **permanent death**(the player loses for good, **no second chances**).

The project must offer the **medium difficulty gameplay** with the original **objective: surviving** a long track back to a main road after getting lost in the woods. The **mechanics** ought to include **sleeping, crafting, item manipulation, collecting resources, travel, fire starting and traps**. The **time progression, status bars and items** should be relatively **similar** to the original.

## Approach

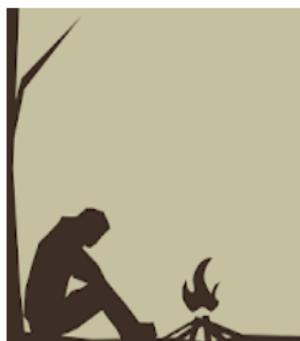
I chose to **implement** the game using **Python**, as I am very comfortable with it.

I tried to use **PyGame**[5] for the **GUI**, until I realized that it is not suited for **mobile applications**, so I switched the progress to **Kivy**[6], a cross-platform Python framework.

I approached **loading/saving** with the Python module **Pickle**[7] and the Android **executable** was made with **Buildozer**[8].

## Related Work

**Survive - Wilderness survival**[1](*Fig.(a)*) is a **mobile game** originally developed by **Juuso Hietalahti**[2]. It has more than 5 million downloads and a 4 star rating from over 127.000 people. The main genre is **Survival**, mixed with **Strategy, Crafting** and **Simulation**. It has an **age restriction** for people under 10 years old. The engine used is **Unity** and the language is **C#**.



Survive - Wilderness survival

Juuso Hietalahti Simulare

★★★★★ 127.338

5 mil.+

E Toți peste vîrstă de 10 ani

(a) App Store Game Banner

**Juuso Hietalathi**[2] is a **game developer** from **Finnland**. He started working on the project during a 24 hour Game Design Contest, where he **won first place** for his **idea** and **rough implementation**. It was so successful that he **continued development** with the intention of **selling** it. Many years later, the **game**(*Fig.(b)*) has **millions of downloads** and there is an **Android** and **iOS** **remake** in the works.



(b) App Store Game Cover

# Contribution

The **thought process** leading to my **choice of project** contained many questions and requirements. It had to be a **game** in my favourite genres, **reasonable to implement** in one **semester** and full of **learning** potential.

My decision, "**Survive - Wilderness survival**" turned out to have many **positives**. Almost **everything** is **new to me** in terms of implementation: the **platform**, the **genres**, the **gameplay**. I have never made an **Android Application** before, so I got to **experiment** with many new routes of **game** design and development. It was very **versatile** in terms of **languages** and **game engines**, so I could choose to be as **comfortable** or as **challenged** as I wanted to be.

This **game** is **not new** to me. I **played** it before, it's the simplest **survival game** I have ever enjoyed. I had **tester insight**, I got to be the **target audience** for the project I was implementing. My **contribution** to the concept can be found in all the **differences** between the **original** and **my** development **journeys**.

## Time Frame

I did not spend **two years** designing a game from **scratch**, I spent **three months** **guessing** the steps someone else made, **analysing** why they made them and why they **succeeded**. I had a lot less **time** to complete a different set of **tasks**.

## Design Stages

My **design decisions** were made as a result of a **two-step process**. **Replaying** the original game as many times as possible in order to **gather data** and guess what values were originally used, and **comparing** it to what I could or what I would rather **prefer** doing.

## Technologies

The **original game** is implemented in **Unity**(a game engine) with **C#**. I used **Python** and a **library**, leaving me a lot of **GUI elements** to make from **scratch**. This helped me a lot in my goal to **learn** as much as possible about the **basics**.

The **original developer** **designed** a game from **scratch**, with an **engine** that allowed a lot of freedom and **readily** implemented code, in the span of **2 years**. I took the role of

**data analyst, tester, target audience and core back-end programmer** for three months, in a **different language**.

# Game Design

## 1.1 Methodology

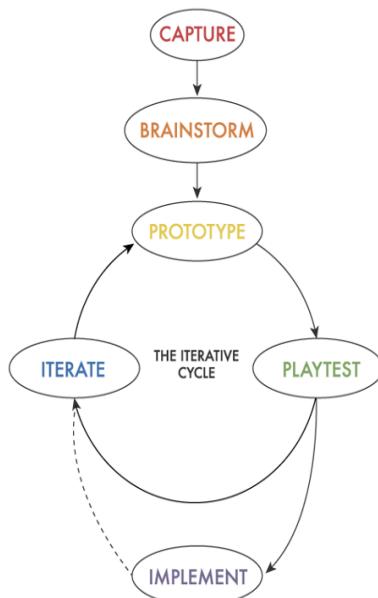
My approach resembled **Iterative Game Designing** the most. Trying new things, testing, discarding and completely changing directions was the only way I could advance towards my main goal, as I was implementing an already existing game just by guessing how it works.

I used a different language and no engine. The first 3 weeks of the development were centered around **PyGame**, a cross-platform collection of modules for User Interfaces. I chose it because I had experience with it and I was comfortable. Unfortunately, it was not suited for **Android** applications, so I had to discard everything and start from scratch with **Kivy**, a framework I had never used before.

I was constantly trying to juggle the **back-end** and **front-end** integration, I couldn't tell how many **iterations** I would need and I had no way of **predicting** many problems in **advance**.

I believe that this **methodology** made me aware of what I don't know. I didn't stick to a topic's **specifics** and I didn't force myself to stop and **plan ahead** too much. It was **less rigid** and **more freeing** and **experimental**.

As a result, I am much more **aware** of what **technologies exist** and how they **intertwine** to **solve a problem**, even if I don't know how they **work**. It taught me what my **minuses** are, what **kind of problems** exist and **where to look** for answers.



(a) **Iterative Game Design**

## 1.2 User Interface

The **design** of the **User Interface** is meant to resemble the **original** one in terms of screen layout, but the design and some button placements were changed to suit my preferences and the resources available.

### 1.2.1 Screen Navigation Architecture

The player should be able to have the same screen navigation experience as it would with the original game.

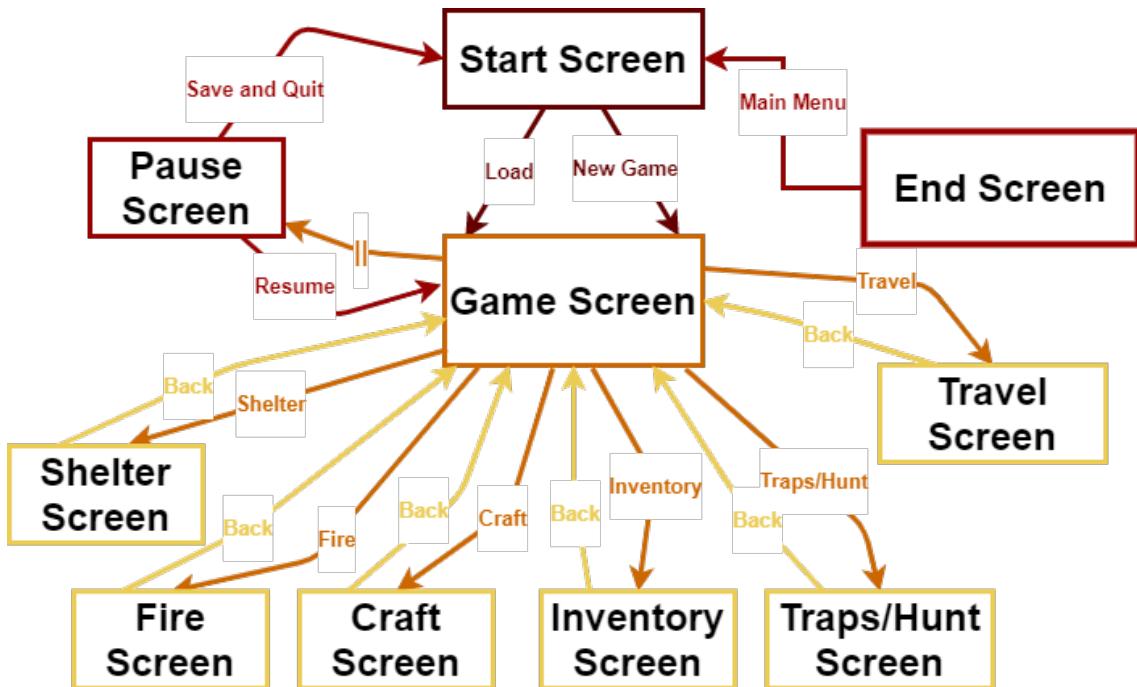


Figure 1.1: Screen Navigation

### 1.2.2 Screens

The entire game can be played with a simple **screen press**, no scrolling needed (including the inventory). There are **Pop Up** screens with **instructions** and **explanations** when necessary.

## Main Menu and Pause Screens

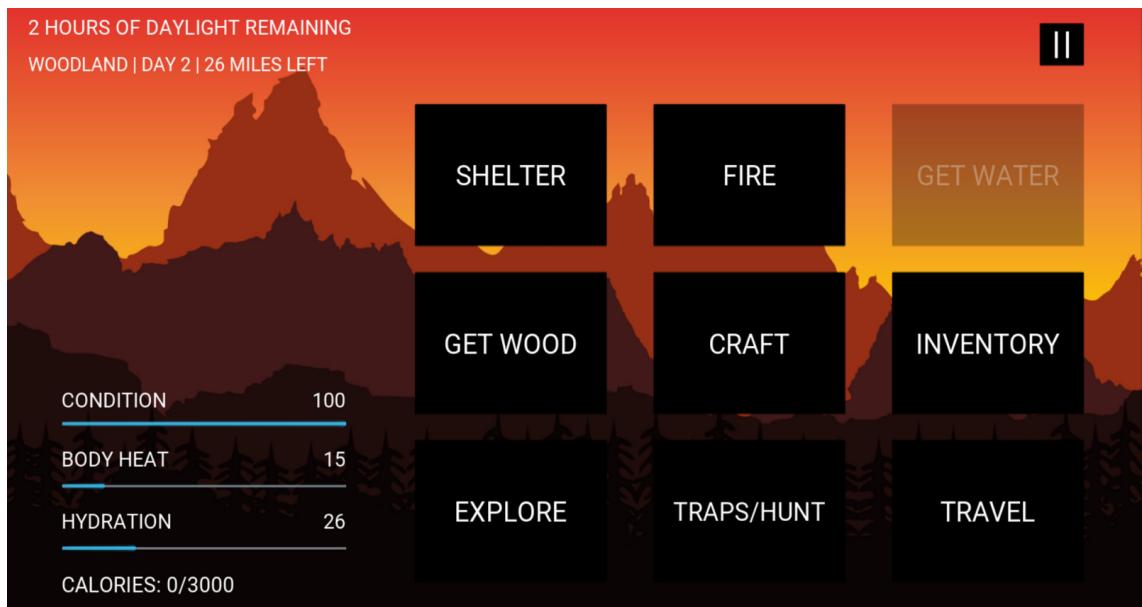


(a) Main Menu Screen



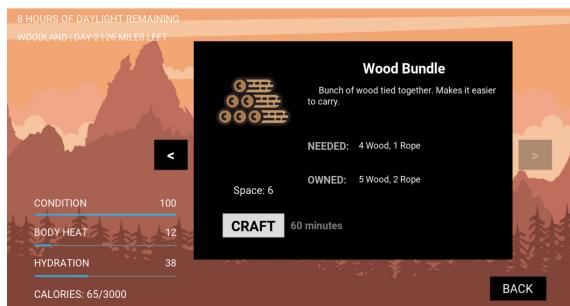
(b) Pause Screen

## Game Menu Screen

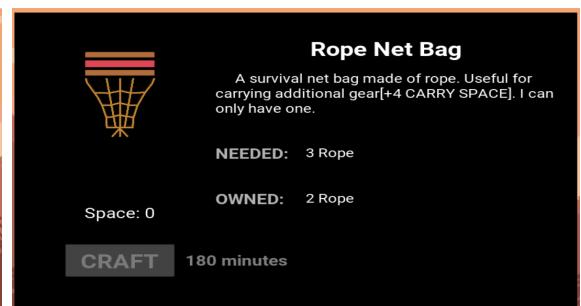


(c) Game Menu Screen

## Crafting Menu Screen



(d) Crafting Menu Screen 1



(e) Crafting Menu Screen 2

## Shelter and Traps Screens

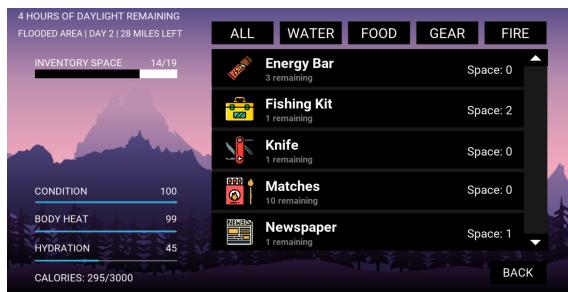


(f) Shelter Screen



(g) Traps Screen

## Inventory Screen

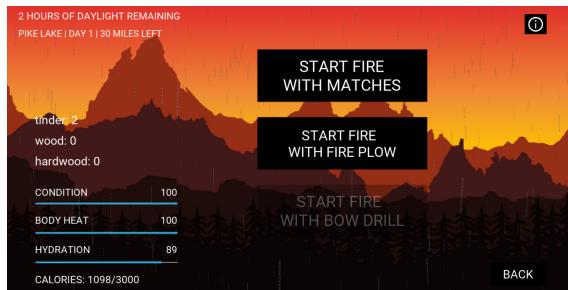


(h) Inventory Screen 1

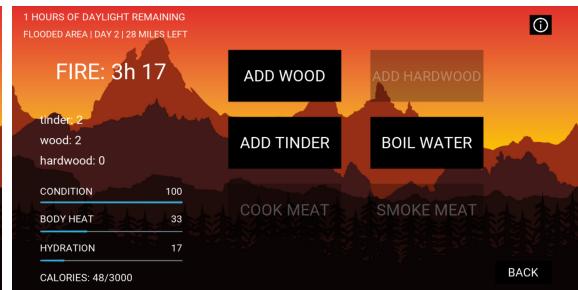


(i) Inventory Screen 2

## Fire Screen

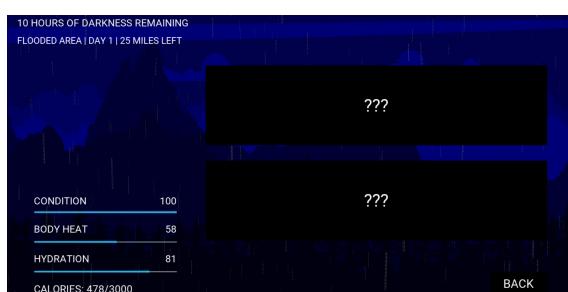


(j) Fire Screen 1

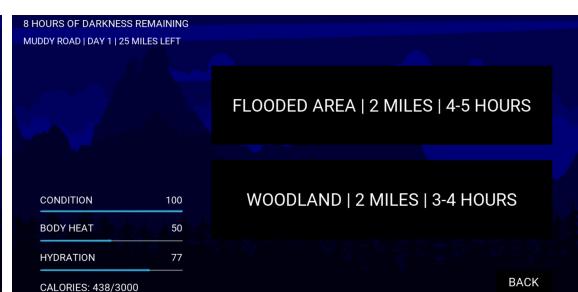


(k) Fire Screen 2

## Travel Screen



(l) Travel Screen



(m) Travel Screen with Flashlight

## End Screen



(n) Win Screen



(o) Loss Screen

# Gameplay

## 2.1 Objective

The main goal is **SURVIVING**: away from the **forest** and back to **civilization**.

### 2.1.1 Environment

The **player** starts the game and immediately gets bombarded with **information** about its surroundings. At the **Pike Lake**, in the **middle of the woods, early afternoon**, during **heavy rain** that has covered the car tire tracks...with no easy way out in sight. It has no choice but to find a way to **survive**.

The **time, day, location, status bars, weather** and **remaining miles** to safety are visible on the **main screen**, ready for strategy planning.

### 2.1.2 Time

The **game time** is one hour for every 3 real minutes. Instant actions such as **sleeping, travelling, crafting** and **exploring** take 2 seconds for any specified amount of time required for them.

The game is designed to offer the player enough time to **think, strategise or peruse** the inventory while taking no time in **advancing the story line**. Just like chess, an advanced "**speedrun**" can be completed in 5 minutes, while a **thought-out**(but poorly played) round can take up one hour and a half with no bore or frustration from the player.

### 2.1.3 Weather

At the beginning, the **main character** is secluded by **rain**. The screen displays a ".gif" raindrop file and any action away from the car can swiftly lead to an **abrupt ending**. The car acts as a **shelter** that allows the game to still **start strong** and create the sense of urgency in order to set the **mechanics, objectives and priorities** for the player.

The lake is the **only** location that completely removes the **body heat depletion**. After leaving the lake, **rain** will be the worst enemy against avoiding hypothermia and death. The user will learn that **sleeping** with surrounding warmth and **preparing for travelling** the next day is the best way to spend the night. Another tutorial-like benefit is showcasing that **clean, safe water** can be collected using a **raincatcher**, while the locations' existing sources are **dirty**.

The **weather** also acts as an incentive to **explore** the very first day, which serves many purposes: teaching the **mechanic**, showing that **resources** are **not limited**, exemplifying the **status bar loss** during actions and making the player realize that Pike

Lake **resources** are exclusive to it and extremely **important** to have for the rest of the **gameplay**. A **raining session** is anywhere between 8 and 72 hours. The **cool down** is 5 to 24 hours.

#### 2.1.4 Day Time

The best time to **travel** and execute any **action** away from the current base is while it's **sunny**. There are **24 hours** in a game day, 12 of daylight of 12 with darkness.

The player **can't** see the next possible **travel locations** without a **flashlight**. Trying to **explore**, **collect wood** and **set traps** will mainly result in **failure**. A flashlight can only be obtained while exploring at the Pike Lake. The item can **never** be found again after that and may even **break**.

**Heat** depletes rapidly at **night**, especially during **rain**, a combination that will mostly result in game **loss**. The best night strategies are keeping a **lit fire**, **sleeping**, **crafting**, using **inventory items** and **planning to take off** first thing in the morning.

#### 2.1.5 Shelter

There is **warmth** from the **car** at the beginning of the game. Future locations require **fire** to at least stop the **heat** from dropping, which is next to that area's **place of rest** and **settlement**, also known as the **shelter**.

The player may **sleep** the nights away or **build a raincatcher** from a **trash bag** in order collect **clean water**.

#### 2.1.6 Locations

The **initial location** of the player is at **Pike Lake**. If it travels, it can **never return** again, as it is headed to the main **road**, in the opposite direction.

There are 5 other **location types**(*Table 2.1*) that serve as an **environment template**. The **player** may encounter the same kind of area **multiple times**, each with different faults and benefits. One may offer **water**, **certain resources** and **animals**, while the **road** to another could be **walked faster**.

Locations	Pike Lake	Flooded Area	Muddy Road	Muddy Area	Path	Woodland

Table 2.1: Game Locations

## 2.2 Status Bars

There are 4 **values** to keep an eye on: **Condition**, **Body Heat**, **Hydration** and **Calories**. All of them can go **below 0**, signifying that the player is on its last breath. **Calories** do not lead to **death**, as a person can **realistically** go weeks **without food**, but the **game ends** when any of the others reach -15%.

## 2.2.1 Condition

**Condition** is a value that can be mostly **left alone**. It improves slightly during the day if it's not maxed out already. Its main purpose arises in times of desperation.

The player may find itself in a **dire situation**, with only **dirty water**, **nearing death thirst** and **no way** to swiftly **start a fire**. Drinking it would offer a way out **once or twice**(*Table 2.2*), but more is mainly **luck**. **Spoiled meat** could offer the necessary calories to travel the **last miles** fast as to not die of thirst, **raw meat** could be near **expiring**.

There is a slight **timely increase**, as well as **bandages** to fix **travel** and **hypothermia wounds**, but **Condition** is mostly an end game **gamble**.

Item	Bandages	Raw Meat	Spoiled Meat	Unsafe Water Bottle
Condition	20	[-70, -30]	[-90, -50]	[-60, 0]

Table 2.2: Values of Condition Changing Items

## 2.2.2 Body Heat

There are 5 factors that influence **heat depletion**: **Shelter(S)**, **Rain(R)**, **Day Time(D)**, **Clothing(C)** and **Fire(F)**.

If the current location is **Pike Lake**, the player only **loses heat** if it **explores** or **travels**. That is the **only** time **shelter** is available. It can be **raining**, it can be **day or night** and the **fire** can be **lit** or not. The player **starts off** with a set of **clothing**, but it can be torn for **pieces of cloth**.

The **table** below(*Table 2.3*) shows all the remaining **factor combinations** and the **values** they influence **Body Heat** with every **hour**. Each "N" preceding a factor letter(mentioned above) signifies "**not**", so "NS" would be no shelter. The code "**NSRNDCF**" reads as "**no shelter, rain, no day(night), clothing(player has it), fire(is lit)**" and depletes **heat** by **-3** for each **hour** that the game is in that state.

NSRDCF	NSNRDCF	NSRNDCF	NSRDNCF
-2	30	-3	-4,5
NSRDCNF	NSNRNDCF	NSNRDNCF	NSNRDCNF
-3,5	25	28	0,5
NSRNDNCF	NSRNDCNF	NSRDNCNF	NSNRNDNCF
-4,5	-8	-6	23
NSNRNDCNF	NSNRDNCNF	NSRNDNCNF	NSNRNDNCNF
-4,5	0	-9,5	-6

Table 2.3: Game Hourly Heat Loss Codes and Values

### 2.2.3 Hydration

**Hydration** is the most **obvious** and **mediocre** status to look after. The player **starts** with several **bottles** of liquid and it can also collect more using **empty bottles** found at **Pike Lake**.

**Building a raincatcher**, collecting from **water sources**(Pike Lake, Muddy Areas and Flooded Areas) and finding **edible berries**(*Table 2.4*) exploring are the ways to **get water**.

Item	Safe Water Bottle	Unsafe Water Bottle	Squirrel Juice	Soda Bottle	Edible Berries
Hydration	25	25	25	25	5

Table 2.4: Values of Hydration Changing Items

### 2.2.4 Calories

The player **can't** die from **calorie loss**, but they are needed for **travelling**, as going below 400 adds 2 extra hours to the **length** of the journey.

Item	Energy Bar	Crickets	Maggots	Soda Bottle	Edible Berries	Squirrel Juice
Calories	150	100	50	300	100	100

Table 2.5: A. Values of Calorie Changing Items

Item	Smoked Jerky	Can of Peas	Edible Plant Part	Raw Meat	Spoiled Meat	Cooked Meat
Calories	300	300	50	400	200	800

Table 2.6: B. Values of Calorie Changing Items

## 2.3 Inventory

The **Inventory** has a **space**(not weight) **capacity** that be increased with **extra carry items** that are crafted or owned from the beginning. If the **space limit** is exceeded, then the player **can't travel**.

There are **interesting ways** of going around this, such as **throwing stuff out**(the cellphone is useless and take 1 space point), crafting **compact items** from more voluminous ones(Wood Bundles), crafting **carry items**(Rope Net Bag, Pouch) and keeping **destructibles intact** until the **resulting resources** are needed, as they usually take **more space**.

### 2.3.1 Items

The **Inventory** holds all of the player's items and allows their **manipulation**(Eating, Drinking, Throwing Out, Destruction).

There are 5 categories: **All(A)**, **Gear(G)**, **Fire(Fi)**, **Food(Fo)** and **Water(Wa)** for **easy navigation**, as seen in Table 2.7.

Item	Space	Throw	Category	Items Obtained From Destruction
Area Map	0.0	Yes	A, G, Fi	Tinder
Backside Of A Paper	1.0	Yes	A, G	Tinder
Bait	0.34	Yes	A, G	-
Bandages	1.0	Yes	A, G	Tinder
Basic Clothes	0.0	No	A, G	Piece of Cloth, Torn Clothes
Birch Bark	1.0	Yes	A, Fi	Tinder
Bow Drill	2.0	Yes	A, Fi	-
Broken Cellphone	1.0	Yes	A, G	-
Can of Peas	1.0	Yes	A, Fo	-
Cattail Plant	1.0	Yes	A, Fo	Edible Plant Part, Plant Fiber, Tinder
Cooked Meat	1.0	Yes	A, Fo	-
Crickets	0.34	Yes	A, Fo	Bait
Dead Bird	3.0	Yes	A, Fo	Raw Meat
Dead Fish	1.0	Yes	A, Fo	Raw Meat
Dead Hare	4.0	Yes	A, Fo	Raw Meat, Bait, Hare Skin
Dry Sack	0.0	Yes	A, G	-
Duct Tape	0.0	Yes	A, G	Rope
Edible Berries	1.0	Yes	A, Fo, Wa	Bait
Edible Plant Part	1.0	Yes	A, Fo	-
Empty Bottle	1.0	Yes	A, Wa	-
Empty Can	0.0	No	A, G	-
Energy Bar	0.0	No	A, Fo	Bait
Fire Plow	1.0	Yes	A, Fi	-
First Aid Kit	2.0	Yes	A, G	First Aid Book, Bandages, Pouch
First Aid Manual	1.0	Yes	A, G	Backside of Paper
Fishing Hook	0.34	Yes	A, G	-

Fishing Kit	2.0	Yes	A, G	Empty Can, Rope, Fishing Hook
Fishing Rod	3.0	Yes	A, G	Wood, Rope, Fishing Hook
Flashlight	2.0	Yes	A, G	-
Hardwood	4.0	Yes	A, Fi	Wood
Hare Skin	1.0	Yes	A, G	-
Knife	0.0	No	A, G	-
Maggots	0.34	Yes	A, Fo	Bait
Matches	0.0	Yes	A, Fi	-
Newspaper	1.0	Yes	A, G	Tinder
Piece of Cloth	1.0	Yes	A, G	Bandages
Plant Fiber	1.0	Yes	A, G	Rope
Pouch	0.0	No	A, G	-
Raw Meat	1.0	Yes	A, Fo	Bait
Rope	0.34	Yes	A, G	-
Rope Net Bag	0.0	Yes	A, G	-
Safe Water Bottle	2.0	No	A, Wa	Empty Bottle
Smoked Jerky	1.0	No	A, Fo	Bait
Soda Bottle	2.0	No	A, Fo, Wa	Empty Bottle
Spoiled Meat	3.0	Yes	A, Fo	-
Squirrel Juice	1.0	No	A, Fo, Wa	Empty Bottle
Tinder	0.34	Yes	A, Fi	-
Torn Clothes	0.0	No	A, G	-
Trash Bag	0.0	Yes	A, G	Rope
Unsafe Water Bottle	2.0	No	A, Wa	Empty Bottle
Utility Belt Bag	0.0	Yes	A, G	-
Wires	0.34	Yes	A, G	Rope
Wood	3.0	Yes	A, Fi	Tinder
Wood Bundle	6.0	Yes	A, Fi	Wood
Wooden Spear	1.0	Yes	A, G	Wood

Table 2.7: Inventory Items

### 2.3.2 Spoilage

There are certain items (*Table 2.8*) that can spoil. Trap prey only lasts a couple of hours after death, when it turns into Spoiled Meat.

The only meat **preservation** method is drying it at the **fire** to make **Smoked Jerky**.

Item	Raw Meat	Cooked Meat	Dead Hare	Dead Fish	Dead Bird
Hours Until Spoilage	15	24	24	8	14

Table 2.8: Items That Spoil and Freshness Durations

## 2.4 Fire

The **fire** screen has the most **complex** process from the **player's perspective**(*Figure 2.1*). It is the **best asset** against **rain** and **night temperatures**.

The **best strategy** is collecting **wood** during the **day** and using it to start a **fire** at **night**, when the player can **sleep** or **prepare** to go to the **next location** at dawn.

**Fire** allows for **water boiling** and **meat processing**, such a **cooking** and **drying** to avoid **Condition loss**.

There needs to be **tinder** and one type of **tool** to **start a fire**. It is **maintained** with **tinder**, **wood** or **hardwood**.

### 2.4.1 Fire Starting

The **fire starting tools**(*Table 2.9*) have a fire starting **success rate**(the **Matches** work all the time, the **Fire Plow** works in 30% of the **cases**). They can also be **destroyed** while being used(the **Matches** are completely used up, the **Bow Drill** has a 3% chance of **breaking**).

Fire Tools	Success Rate	Destruction Rate
Matches	100%	100%
Bow Drill	50%	3%
Fire Plow	30%	6%

Table 2.9: Fire starting Tools and Quality

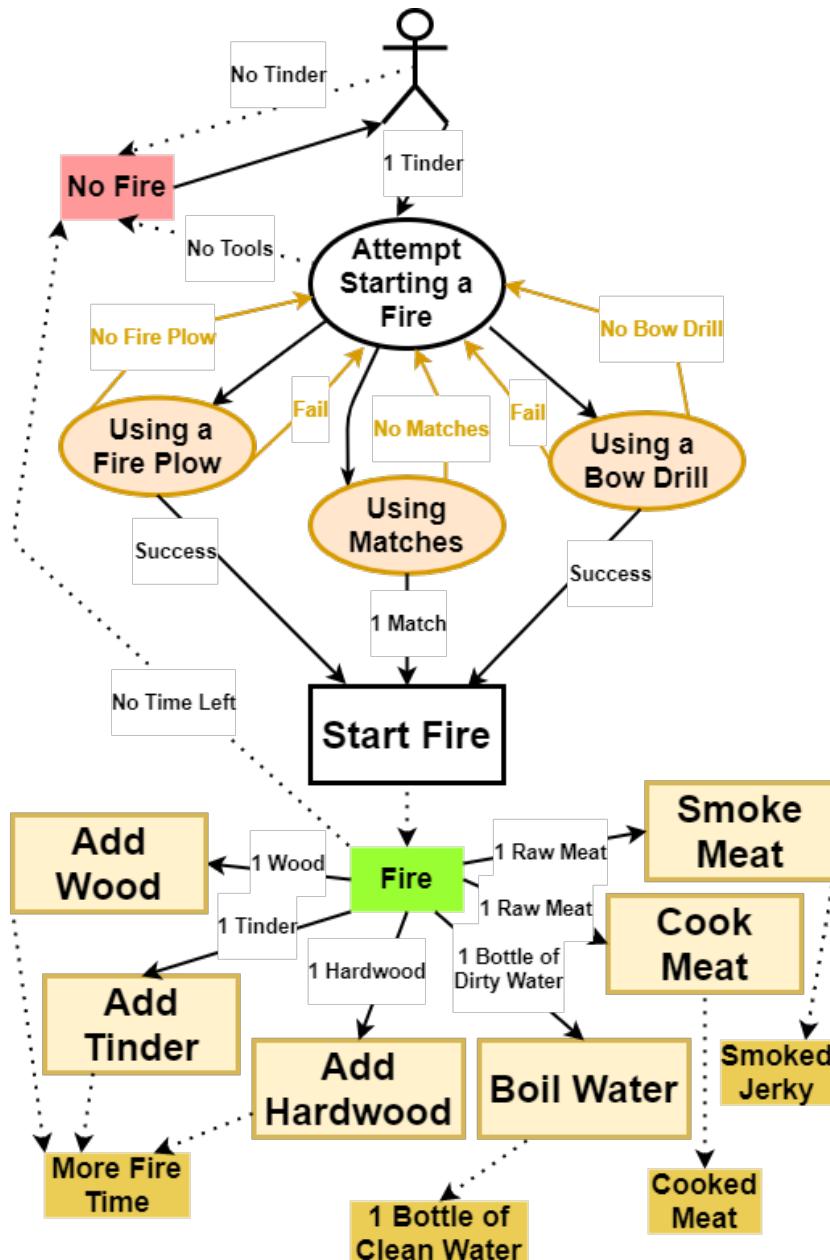


Figure 2.1: Fire Screen Process

## 2.5 Crafting

Creating **complex** items is displayed as **crafting**, while **breaking** them apart to get **parts** can be done in the **inventory**. Making something new takes **time** and **place** near the **fire**, so it can be done at **night** with no **Body Heat** loss.

The table below (*Table 2.10*) showcases all the **items** that can be **created**, the **time** needed, the **resources** and their **descriptions**. There can **only** be **one** of a certain device in the **inventory**(Fire Plow, Bow Drill).

Crafting can help with **Inventory Management** in many ways. There are **bags** for extra space(Pouch) or **creations** that are **more compact** than their **loose parts**(Wood Bundle, Rope).

Item	Crafting Duration	Crafting Resources	Description
Fire Plow	30 minutes	Wood	"A primitive tool for starting a fire. Rubbing two sticks together should produce an ember. I can only have one."
Bow Drill	30 minutes	Rope, Hardwood	"A primitive tool for starting a fire. Using the bow drill requires some skill and plenty of patience. I can only have one."
Fishing Rod	30 minutes	2 Ropes, Fishing Hook, Wood	"A survival fishing rod made of stick, rope, and a hook. Good enough to catch a fish. I can only have one."
Wooden Fishing Hook	60 minutes	Wood	"A fishing hook. Sharp end."
Rope	30 minutes	Piece Of Cloth	"Piece of rope. Useful for many purposes in a survival situation. Each piece of rope requires 1/3 unit of carry space."
Wooden Spear	30 minutes	Hardwood	"It's a wooden spear with a pretty sharp tip. Better than bare handed hunting. I can only have one."
Rope Net Bag	180 minutes	3 Ropes	"A survival net bag made of rope. Useful for carrying additional gear [+4 CARRY SPACE]. I can only have one."
Pouch	120 minutes	Hare Skin, Rope	"A small pouch. I can carry stuff in it. I can have a couple of these. [+1 CARRY SPACE]"
Wood Bundle	60 minutes	4 Woods, Rope	"Bunch of wood tied together. Makes it easier to carry."

Table 2.10: Craftable Items

## 2.6 Location Specific Actions

### 2.6.1 Exploring

Each time the player arrives somewhere new, the game takes the location type and uses its "Number of Resources" interval (*Table 2.11*) to randomize how many resources to create.

Location	Pike Lake	Flooded Area	Muddy Road	Muddy Area	Path	Woodland
Number Of Resources	{3, 6}	{3, 6}	{9, 11}	{7, 9}	{8, 10}	{6, 8}

Table 2.11: Amount of Resources at a Location Type

Each **location type** has some **items** and their **probability to be randomized** in the **current** place's "Available Explorables" list (*Table 2.12*).

Resource/Location	Flooded Area	Muddy Road	Muddy Area	Path	Woodland
Wood	-	20%	35%	-	-
Cattail Plant	-	-	-	24%	28%
Maggots	17%	-	-	29%	38%
Crickets		14%	23%	33%	28%
Cattail Plant	45%	-	-	-	-
Edible Berries	-	22%	12%	14%	6%
Plant Fiber	38%	9%	23%	-	-
Birch Bark	-	24%	7%	-	-

Table 2.12: Chance of Item Finding at Certain Locations

**Pike Lake** has some **exclusive** items that are **very useful** for the **gameplay**, as it can only be visited **once**, at the beginning. (*Table 2.13*).

Item	Flash Light	Wires	Empty Bottle	News Paper	Duct Tape	Fishing Kit	Matches	First Aid Kit
%	15%	25%	10%	15%	9%	8%	8%	10%

Table 2.13: Chance of Finding an Item at Pike Lake

## 2.6.2 Getting Wood

**Wood** can't be collected at **Pike Lake**, in order to promote **Exploring** for **new players** and because the **car** acts as a **shelter**. There is **no limit** at any locations, the player only has the **night** without a **flashlight** to worry about.

## 2.6.3 Getting Water

**Clean water** can be obtained through a **raincatcher**, but **dirty water** can be collected from locations that have it as a **resource** (*Table 2.14*).

Location	Pike Lake	Flooded Area	Muddy Road	Muddy Area	Path	Woodland
Dirty Water Source	Yes	Yes	No	Yes	No	No

Table 2.14: Location Types Water Resources

## 2.6.4 Setting Traps and Catching Fish

Traps have their own screen that allows building and dismantling. The amount that can be made of each is infinite if the resources are there, so the chance gets better with each new one(*Table 2.15*). Fish can be caught with a spear(5%) or fished(8%).

Traps	Hourly Trap Rate	Prey	Prey Resources	Building Resources
Deadfall	2%	Hare	3 Raw Meats, Bait, Hare Skin	Rope, Wood, Bait
Fish Trap	3%	Fish	1 Raw Meat	Rope, Wood, Bait
Bird Trap	4%	Bird	2 Raw Meats	Rope, Wood, Bait

Table 2.15: Traps

## 2.7 Travel

Travelling to a location requires an inventory below the space threshold. The number of travel hours is randomized from the specific interval.

If the player has less than 400 calories, 2 extra hours are added to the trip. The location choices are not visible at night without a flashlight(*Table 2.16*).

Location	Miles	Hours	Water Source
Pike Lake	3	{4, 5}	Yes
Flooded Area	2	{4, 5}	Yes
Muddy Road	2	{3, 4}	No
Muddy Area	2	{3, 4}	Yes
Path	3	{3, 4}	No
Woodland	2	{3, 4}	No

Table 2.16: Travel Information

# Implementation

## 3.1 Technologies

### 3.1.1 Kivy

Kivy[6] is a **cross-platform framework** that allowed me to build the game as an **Android application**. The relationship between **Kivy** and **Python** reminds me heavily of the one between **CSS/HTML** and **PHP**. Placing **Widgets** in **".kv"** files feels like building a **web site**. My **text-based, screen-heavy** game took **thousands** of lines of **Kivy** code.

### 3.1.2 Pickle

Pickle[7] is a **Python module** used for **serializing object structures**. I used it to implement **saving, loading and pausing** on a **Game State object** that holds all the **information** about the **gameplay**.

### 3.1.3 Buildozer

Buildozer[8] is a tool that packages **mobile applications**. I used it to create the **Android executable** for the game. It requires **Linux** for compilation, so I shared the code between **Ubuntu** and **Windows** via **Github** in order to use it.

## 3.2 Integration

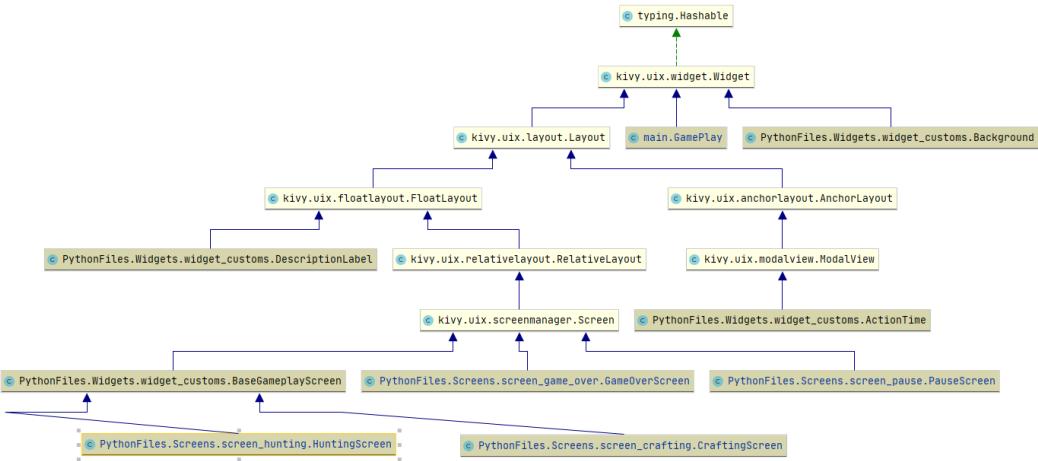


Figure 3.1: Piece of Kivy and Python Integration Architecture

Python makes use of **Kivy Widgets** in order to create **UI elements**. This can be seen in *Figure 3.1*, which contains a piece of **integration architecture**(with most elements discarded to fit).

### 3.2.1 Kivy Using Kivy Files

Kivy code can be written in both `".kv"` and `".py"` files. *Listing 3.1* displays a piece of code that adds a **Button Widget** and an **Image Widget**(a "pause" symbol) to three selected **screens**.

The language's **structure** resembles **HTML**. It allows for compact code as **multiple screens** can share the same **Widgets**. The `"root.show_info()"` line calls the `"show_info()"` **method** of the **parent object** found in a `".py"` file. The `"on_release"` line allows behaviour addition and makes the button **responsive**. The **image** is added on top and does not **interfere** with the **functionality** of the **button**.

```
1 <FireScreen , HuntingScreen , ShelterScreen >:  
2     FloatLayout:  
3         Button:  
4             pos_hint:{ "x": 0.91 , "y": 0.90 }  
5             size_hint: 0.04 , 0.07  
6             background_color: 0.0 , 0.0 , 0.0 , 1.0  
7             bold: True  
8             font_size: self.height*0.5  
9             text: ""  
10            on_release:  
11                root.show_info()  
12            Image:  
13                source: "GraphicFiles/info.png"  
14                size_hint: 0.026 , 0.054  
15                pos_hint:{ "x": 0.917 , "y": 0.908 }  
16                allow_stretch: True
```

---

Listing 3.1: Kivy Implementation of the Pause Button

### 3.2.2 Kivy Using Python Files

Widgets can be implemented in ".py" files as well. "ModalView" creates a Pop Up the size of the current window. The code in *Listing 3.2* implements a "ModalView" that displays an "action execution" text and closes after a second("Building...", "Sleeping...").

```
1 class ActionTime(ModalView):
2
3     def __init__(self, text, **kwargs):
4         super(ActionTime, self).__init__(**kwargs)
5
6         # Set the specific text for the action
7         self.ids.action_time_label.text = text
8
9         # Call dismiss_view after one second
10        Clock.schedule_once(self.dismiss_view, 1)
11
12    def dismiss_view(self, dt):
13        self.dismiss()
```

Listing 3.2: Kivy Implementation of the Action Execution Pop Up

*Listing 3.3 showcases* the piece of code that changes the background picture to reflect the time of the day.

```
1 class Background(Widget):
2     """ Class for changing the background during the day """
3
4     # The path to background image
5     this_source = StringProperty("GraphicFiles/night.png")
6
7     def change_background(self, *args):
8         """ Change background after the day period """
9         # Total game hours that have passed
10        game_hours = init.game_state.game_time / 60
11        # The current day of the game
12        init.game_state.current_game_day = int(game_hours / 24 + 1)
13        # Game hours that have passed in the current day
14        game_hours_today = game_hours % 24
15
16        # Change background image path based on the time of the day
17        if game_hours_today < 2:
18            self.this_source = "GraphicFiles/dawn.png"
19        if game_hours_today < 7:
20            self.this_source = "GraphicFiles/orange.png"
21        elif game_hours_today < 10:
22            self.this_source = "GraphicFiles/purple.png"
```

```

23     elif game_hours_today < 12:
24         self.this_source = "GraphicFiles/sunset.png"
25     else:
26         self.this_source = "GraphicFiles/night.png"

```

Listing 3.3: Kivy Implementation of the **Background Change**

## 3.3 Testing

The **project** was **manually** tested after each **iteration**. The nature of the game allows for extremely **fast playthroughs**, as each action takes **one second** in real time. This was a **best case** scenario, as each feature could be **tested** in a matter of at most **several minutes**. Testing **automation** was not really needed.

### 3.3.1 Quality Assurance

As far as I am concerned, there are **no bugs** remaining. The addition of the **rain ".gif"** **animation** makes the **game** take a minute to **open** on **mobile devices** after the first installation, but the application is very **responsive** otherwise. "**Speedruns**" can cause **occasional** lags of 2-3 seconds, but **fast gameplay** poses no problems.

### 3.3.2 Playtesting

The **application** was **tested** by friends, family and other students. As I was implementing an already **successful game**, I didn't really have many **difficulties** with the **mechanics** and **gameplay design** choices. However, there are things that I **changed** to **my liking** and I made sure to **cater** to all the **complaints** and **suggestions**.

# Conclusion

Making this **game** was a very **gratifying** and **fulfilling** experience. I succeeded to **implement** all of the **mechanics** and **features** of the original game's **medium difficulty** gameplay. The application can be **downloaded** and **played** on **Android** mobile devices.

The **user** can **pause**, **resume**, **save** and **quit** the game with no negative impact on the **time progression** and the **permanent death** feature. The **environment** has all the original **locations** and **resources**. **Sleeping**, **crafting**, **travel**, **traps** and **hunting** were also fully implemented. All the **inventory items** can be manipulated and used in the intended ways. **Exploration** and **resource gathering** have similar, if not identical chance **values**.

The **graphical interface** has **bright colours** and **beautiful backgrounds**. The **screen structure** is similar, but the **design** was tweaked to my liking. I did not, however, manage to implement a **sound** system.

I chose to make the **game** in **Python** because I have done this before and I was **comfortable** with it. I am glad I took the time to make a lot of the **GUI** elements from **scratch** and learn about **integration**, but I will not do this again. I cherish the **knowledge**, but I have gained enough **confidence** to use **game engines** from now on.

In **conclusion**, all of the primary **objectives** of the **project** have been **achieved**.

# Bibliography

- [1] Survive - Wilderness Survival  
<https://play.google.com/store/apps/details?id=com.sandbaygames.survive&hl=ro&gl=US>
- [2] Juuso Hietalahti  
<https://fi.linkedin.com/in/juusohietalahti>
- [3] Game Design Course  
<https://profs.info.uaic.ro/mmoruz/courses/GD2021.html>
- [4] Iterative Game Design Diagram  
<http://www.backyardgamedesign.com/blog/2015/5/11/game-design-process>
- [5] Pygame Documentation  
<https://www.pygame.org/news>
- [6] Kivy Documentation  
<https://kivy.org/#home>
- [7] Pickle Documentation  
<https://docs.python.org/3/library/pickle.html>
- [8] Buildozer Documentation  
<https://buildozer.readthedocs.io/en/latest/>
- [9] Flaticon - Inventory Item Icons  
<https://www.flaticon.com/search?word=wood>
- [10] pngtree - Background Images  
<https://pngtree.com/user/my-favorites?type=back&page=2>