#### Developer’s wishlist

##### V1.0 – V1.2

**Developed by:** Kory

**Focus:** Making it calculate the results

**Release date:** ~ 2018 with a couple of minor updates since then

**Status:** Abandonware. No longer maintaining this

|  |  |
| --- | --- |
| **Implemented** | **Feature** |
| **Storing information about battle participations** | |
|  | **Using struct to store:**   * The name of the entity **(1)** (including civilization) **(4)**   string unitName   * The armour class **(4)**   bool armorClass[20];   |  |  | | --- | --- | | * Archer [0] * Building [1] * Camel [2] * Castle [3] * Cavalry [4] * Cavalry\_Archer [5] * Eagle\_Warrior [6] * Gunpowder\_Unit [7] * Infantry [8] * Monk [9] * Ram [10] | * Ship [11] * Siege\_Weapon [12] * Spearman [13] * Standard\_Building [14] * Stone\_Defence [15] * Turtle\_Ship [16] * Unique\_Unit [17] * Wall\_&\_Gate [18] * War\_Elephant [19]. |  * The attack bonuses vs amour classes **(5)** * The health **(7)**   int unitHealth;   * The ranged damage (RD) value **(8)**   int rangedDamage;   * The standard damage (SD) value **(9)**   int standardDamage;   * The Age pre-requisite of the card **(10)**   int unitAge;   * The point value **(12)**. Sum of resource cost. What’s awarded for killing the entity   int pointValue;   * The garrison value, which appears on some buildings in case it’s ever relevant   int garrisonValue;   * The quantity of the entity   int unitQuantity;  **Example “Crossbowman” card**    Using struct as I can store multiple data types per entity and return this entity. |
|  | Having a blank entity, which can be used to reset the values of a selection |
|  | Having two entities that represent what players 1 and 2 are putting into battle |
| **Applying the effects of modifiers (attack bonuses, event cards, and technologies) as well as the quantity of units** | |
|  | Modifying the values of player 1’s selection and player 2’s selection based on modifiers  Approach works because two targets cannot attack one simultaneously |
| **Calculating the outcome of different round of combat** | |
|  | Calculating the outcome of an archer round of combat (ranged entities may attack and may retreat). The round is negated if fighting cavalry |
|  | Calculating the outcome of the standard two rounds of combat (can retreat after 1 round) |
|  | Calculating the outcome of a monk round of combat |
|  | Calculating the outcome of a bombardment round of combat |
| **Getting information about each player’s “play state”** | |
|  | **Reading info from .csv files:**   * Each player’s entities * The quantity of each player’s entities * Each player’s technologies * Whether these technologies are in play   Not using std::cin for this as there’s too much that would need to be entered and it inserts odd symbols for capitals and underscores https://cdn.discordapp.com/attachments/442575858096668672/442694350489518085/unknown.png |
|  | **Validating the input**   * Converting names to uppercase format. Seeing if the name entered matches one of the accepted names of entities (in uppercase). * Making sure the user fills out all the fields and that there’s not too little or too much input * Making sure the names of entities put in are recognized * Making sure that there are no spaces as I’m using the spaces to split up the fields. * Making sure that the quantity of participating entities (except assisting monks) is > 0 * And < 2 for buildings * And < 6 for all other entities |
|  | Using the info from the .csv files to search for the corresponding thing and filling in the rest of the details so no user input is needed for that   |  | | --- | | **Example** | | if(entityName == “Archer\_(Saracen)”) then{  currentSelection = {“Archer\_(Saracen)”,1,entityQuantity,6,4,5,2,  true,false,false,false,false,false,false,false,false,false,false,false,false,false,false,false,false,false,false}; // One of the player’s entered Archer\_(Saracen)  } | |

##### V2.0

**Developed by:** Kory & Phillip

**Focus:** Converting it from a terminal application into a cross-platform GUI application

**Goal:**

* Making it easier for users to use through the addition of a GUI
* Making it faster for users to perform actions like entering players' battle-relevant information and calculating the results based on this by using hotkeys and UI elements
* Making it easier for users to run it.
  + Running the program via clicking an executable (.exe) file, not via typing "./run" in a Linux shell
* Making it easier for users to install / uninstall through software packaging
* Making it handle the last few event cards I haven't implemented (e.g., *"Back from a Foreign Land"*)
* Making it so users can run it on a range of devices with different operating systems
* Making it more customizable (e.g., being able to set player names)
* Making it prettier

**Status:** In development

|  |  |  |  |
| --- | --- | --- | --- |
| **Implemented** | **Feature** | | |
| **Adding essential functions** | | | |
|  | Exit button | | |
|  | About button | | |
|  | View user guide button | | |
|  | View developer guide button | | |
| **Getting information about the player’s *“play state”*** | | | |
|  | Getting the entity (unit, building) names | | |
|  | Getting the quantities of entities | | |
|  | Getting the quantities of monks | | |
|  | Setting up a filter for the entity names | | |
|  | Making it easier to search for entity names   * Being able to use aliases so “Temple of the Sun at Macchu Picchu (Incan)” can be found by searching for “Wonder (Incan)” * Making it so the capitalization doesn’t matter | | |
|  | **Validating the input:**  Limiting what can be entered for the quantity of entities to 1 - 5 | | |
|  | **Validating the input:**  Limiting what can be entered for the quantity of monks to 0 - 5 | | |
|  | CheckedListBox in QT for events & technologies <https://www.walletfox.com/course/qtcheckablelist.php> | | |
|  | Setting up the initial state of the UI elements with what’s in the .csv files  (Done with exception of player colour) | | |
| **Sending information about the player’s *“play state”* to the files so that the backend can read it.** | | | |
|  | Converting entered entity names into names with underscores between it | | |
|  | Sending what the user has entered to the .csv files  (Done with exception of player colour) | | |
| **Getting user input** | | | |
|  | Supplying variables with answers the user provides. Was using std::cin for this. Perhaps we’d need to use a popup if we’re using QT framework. The QInputDialog class seems to be what we want. Would prefer it if there were like buttons corresponding to the number of options, however | | |
| **Having hotkeys** | | | |
|  | Having a hotkey for the press of the calculate button | | |
| **Displaying output** | | | |
|  | Writing std::cout statements to the GUI | | |
|  | Making it so it clears the existing output when clicking the “Calculate results” button again. Don’t want to see the same output twice or irrelevant output | | |
|  | Making it so the output text is coloured, perhaps using HTML (Partially done) | | |
| **Making it prettier** | | | |
|  | Adding colour to the UI elements   * Having a light and dark mode | | |
|  | Adding an application icon | | |
| **Making it easier to identify the players in the output window** | | | |
|  | Having an option for player name in playerDetails.csv | | |
|  | Having an option to set the player colour  (Partially done) | | |
| **Making it usable on a range of devices of different screen sizes** | | | |
|  | Making it responsive | | |
| **Playing sound effects from Age of Empires 2 (in moderation, don’t want it to be obnoxious)**  [https://docs.google.com/spreadsheets/d/1bczdFQksnbLnjI5zAkw-mSpb9MnnxxEkHDiz1PftIHw/edit#gid=123661276](https://docs.google.com/spreadsheets/d/1bczdFQksnbLnjI5zAkw-mSpb9MnnxxEkHDiz1PftIHw/edit%23gid=123661276) | | | |
|  | | Having AoE II sound effects for the UI | |
|  | | Having AoE II sound effects for RNG elements   * Successful “conversion attempt” * Successful “healing attempt” | |
|  | | Having AoE II sound effects for significant events   * Destruction of a wonder | |
|  | | Having an option to enable and disable SFX | |
| **Covering what I haven’t covered and should have covered in a v1.0 – v1.2 release** | | | |
|  | Add this event card   * **"Back From A Foreign Land":** “Use 1 Civilization bonus from target player for this turn. Bonus may not be used if it is a starting bonus **or requires civilization specific cards**. Play anytime.”. **Really only 3 battle relevant bonuses to choose from in that case** * **Byzantine bonuses to choose from:** * Monk Healing Rate has a +2 modifier, thus making it easier to heal units * All building get a HP bonus: Age I – 10 HP, Age II – 20 HP, Age III – 30 HP, Age IV – 40 HP * **Teuton bonuses to choose from:** * Conversion rate modifier is -1, thus making it harder to convert | | |
|  | **Add this event card**   * **“Holy War”**: "For the next 3 turns (not including your current turn) all of your units get +4 AP during this time. | | |
|  | **Add this event card**   * **“Black Knight”**: “Play this card when you are the attacking Cavalry unit. Two tokens on the defending unit have 0 AP for the first round of normal combat. | | |
| **Fixing bizarre problems (if possible)** | | | |
|  | | | Menu items appearing on second monitor |
| **Making it easier to run and install** | | | |
|  | Deploying it and packaging it with Inno Setup | | |
| **Making it so.docx files are instead opened as PDF files within the application**  [**https://doc.qt.io/qt-6/qtpdf-index.html**](https://doc.qt.io/qt-6/qtpdf-index.html) | | | |
|  | **Implementing a PDF viewer for the user guide, developer guide, and developer wishlist files** | | |

events, technologies

have option to filter all ticked ones

entities - have icons beside names

##### V3.0

**Focus:** Connecting a SQL database to a C++ program

**Goal:**

* Making it easier for designers to add, modify, view, and delete entity (unit, building) values

**References:**

Chapter 7 of “Learn QT 5”. Opting to use this one first as it talks about using SQLLite

<https://doc.qt.io/qt-6/qtsql-index.html>

|  |  |
| --- | --- |
| **Implemented** | **Feature** |
| **Having data about the entities (units, buildings) stored in a SQL database instead of hard coded into the program** | |
|  | Units table |
|  | Buildings table |
| **Supplying the program with this data** | |
|  | Perhaps running a SQL query in C++ and storing the result of this as a variable |
| **Having a developer window** | |
|  | Add new entities |
|  | Delete existing entities |
|  | Modify the entity values |
|  | View the entity values |