**Μάθημα**

Αλγοριθμική Σκέψη

**Εργασία:**

Tanks: Ένα Python Multiplayer Game,

με χρήση της Server – Client Αρχιτεκτονικής

**Υπεύθυνοι Καθηγητές:**

Μηνάς Δασυγένης

Πλόσκας Νικόλαος

**Φοιτήτρια:** Κωνσταντινίδου Αθηνά

***Συνεργάτης:*** *Κοσμάς Σουρλής*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ΠΕΡΙΕΧΟΜΕΝΑ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ΕΝΟΤΗΤΑ 1η**

# ΕΙΣΑΓΩΓΗ ……………………………………………………………………………………….…….…………..….3

# ΓΕΝΙΚΗ ΑΝΑΣΚΟΠΗΣΗ ………………………………………………………………………………….…….….3

# ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ …………………………………………………………………..……..….….3

# ΧΡΗΣΗ & MECHANICS ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ ………………………………………………..…..……….….3

# ΟΔΗΓΟΣ ………………………..…………………………………………………………………..…….………..….3

# ΣΚΟΡ, ΝΙΚΗ & ΗΤΤΑ ……………………………………………………………………………..………...….….3

# ΙΔΙΑΙΤΕΡΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ………………………………………………………………………….…….….3

**ΕΝΟΤΗΤΑ 2η**

# ΤΕΧΝΙΚΕΣ ΛΕΠΤΟΜΕΡΕΙΕΣ …………………………………………….………………………..……...….….3

# ΑΡΧΙΤΕΚΤΟΝΙΚΗ ……………………………………………………………………………………..………….….3

# CLIENT & SERVER …………………………………………………………………….………………........….….3

# ΕΠΕΞΗΓΗΣΗ ΤΟΥ ΚΩΔΙΚΑ & ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑ …………….…………………………..………...….3

# ΠΡΟΚΛΗΣΕΙΣ & BUGS ……………………………………………………………….…………..………….…….3

**ΕΝΟΤΗΤΑ 3η**

# ΙΣΤΟΡΙΚΟ ΥΛΟΠΟΙΗΣΗΣ …………………………………………………………….……………..……......….3

# ΠΕΡΑΙΤΕΡΩ ΕΞΕΛΙΞΗ …………………………………………………….………………………..……..…...….3

# ΣΥΜΠΕΡΑΣΜΑΤΑ ………………………………………………….…..…………………………..…….….....….3

# ΕΥΡΕΤΗΡΙΟ ΟΡΩΝ ……..……………………………………..………………………………..……………...….3

.

**ΕΝΟΤΗΤΑ 1η**

# ΕΙΣΑΓΩΓΗ

Με αφορμή, την εκπόνηση της εργασίας στο μάθημα Αλγοριθμική Σκέψη του ΠΜΣ Ανάπτυξη Ψηφιακών Παιχνιδιών και Πολυμεσικών Εφαρμογών του Πανεπιστημίου Δυτικής Μακεδονίας, υλοποιήθηκε ένα multiplayer network παιχνίδι, στη γλώσσα Python. Το παιχνίδι ονομάζεται ‘’Tanks!’’. Επιλέχθηκε το multiplayer χαρακτηριστικό, καθώς είναι πιο δημοφιλές και στα σύγχρονα παιχνίδια. Επιπλέον, δόθηκε ιδιαίτερη έμφαση στην κατασκευή βάσεων δεδομένων και ανταλλαγή (σε πραγματικό χρόνο) δεδομένων μεταξύ ενός Server και ενός Client, καθώς αποτελεί επίσης πολύ δημοφιλές χαρακτηριστικό στα σημερινά παιχνίδια. Σε αυτήν εδώ την αναφορά, εμβαθύνουμε περισσότερο στα χαρακτηριστικά του gameplay του παιχνιδιού, στις τεχνικές πληροφορίες, στη διαδικασία υλοποίησης του και γενικά σε πληροφορίες σχετικά με την ανάπτυξη του.

# ΓΕΝΙΚΗ ΑΝΑΣΚΟΠΗΣΗ

Βρισκόμαστε στη μέση ενός νησιού (όπως διακρίνεται και από τη οθόνη έναρξης), προστατεύοντας ένα κάστρο, το οποίο είναι το τελευταίο καταφύγιο της χώρας. Όλα τα υπόλοιπα καταφύγια έχουν πέσει στα χέρια των εισβολέων. Προσπαθούμε να πετύχουμε τα εχθρικά τανκς ενώ κινούμαστε γύρω από το κάστρο με το δικό μας τανκ. Σκοτώνοντας τους εχθρούς, προστατεύουμε το κάστρο και αυξάνουμε το σκορ μας.

# ΠΕΡΙΓΡΑΦΗ ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ

Πρόκειται για ένα Multiplayer παιχνίδι, όπου δύο διαφορετικοί παίκτες εφόσον βρίσκονται στο ίδιο δίκτυο Wifi, συνδέονται από δύο διαφορετικούς υπολογιστές. Επιλέγει ο καθένας ξεχωριστά το βαθμό δυσκολίας (Easy, Medium, Hard), το tank που θα έχει και συνδέεται στο παιχνίδι.

Ένα ενδιαφέρον χαρακτηριστικό που υλοποιείται στην εφαρμογή, είναι η αυτόματη επιλογή των επιπέδων και του εναπομείναντος tank, για τον παίκτη που θα επιλέξει δεύτερος, καθώς ο παίκτης που επέλεξε πρώτος χρονικά έχει ήδη αποκλείσει τον δεύτερο από κάποιες επιλογές. Για παράδειγμα, αν ο ένας επιλέξει το μπλε τανκ, τότε αναγκαστικά, η μόνη επιλογή που δίνεται στον άλλο παίκτη, είναι το κίτρινο.

Η αρχική ιδέα του παιχνιδιού ήταν η υποστήριξη πολλών modes. Θα υπήρχε, δηλαδή η δυνατότητα τόσο τα tanks να πολεμούν μεταξύ τους, όσο και ύπαρξη ενός δεύτερου mode. Εκεί, τα δύο tanks συνεργάζονται στην ίδια ομάδα και αντιμετωπίζουν εχθρούς που κάνουν spawn, με σκοπό να επιτεθούν στο κάστρο. Η υλοποίηση του δεύτερου mode με Client-Server αρχιτεκτονική, ήταν αρκετά πολύπλοκη, οπότε η ιδέα αυτή αφέθηκε. Παρόλα αυτά, οι κώδικες για το spawn τέτοιων sprites υφίστανται στον κώδικα, χωρίς όμως να δημιουργούν κάτι, όπως φαίνεται και στο παρακάτω **στιγμιότυπο.**

# ΧΡΗΣΗ & MECHANICS ΤΟΥ ΠΑΙΧΝΙΔΙΟΥ

Το πρώτο που αντικρύζει ο παίκτης με το που ανοίγει το παράθυρο είναι

Παίκτης

Κίνηση παικτών:

Σφαίρες

Κάστρο

Έδαφος

# ΟΔΗΓΟΣ

Απαραίτητες προϋποθέσεις για την εγκατάσταση και την εύρυθμη λειτουργία της εφαρμογής:

# Αρχικά, γίνεται η λήψη του συμπιεσμένου αρχείου zip , τόσο από το [ClientGame Repository](https://github.com/nanakonsta/ClientGame-master), όσο και από το [ServerGame Repository](https://github.com/nanakonsta/ServerGame-master), μέσω github, όπως δείχνουν οι εικόνες:

# Γίνεται

# λήψη

# ΣΚΟΡ, ΝΙΚΗ & ΗΤΤΑ

Στην αρχή, οι παίκτες ξεκινάνε με ένα συγκεκριμένο νούμερο στο health bar τους, το οποίο μπορεί να κατέβει αν τους πετύχει ο αντίπαλος. Όταν το health μηδενίσει, οι παίκτες χάνουν. Μάλιστα υπάρχει διαθέσιμο και το feature του να «πεταχτούν» αυτόματα από το παιχνίδι στο ενδεχόμενο ήττας. Ενώ αν μηδενίσει το health του αντιπάλου (κάτι που επιτυγχάνεται πυροβολώντας τον), τότε νικάνε. Παράλληλα ενσωματώσαμε και καταμέτρηση του σκορ και των πόντων που πετυχαίνουν, παρότι δεν έχει ρυθμιστεί το να εκφράζεται ακόμη.

Υπάρχει δυνατότητα επέκτασης του κώδικα με το health του κάστρου, για μετέπειτα εξέλιξη του παιχνιδιού στο mode, όπου τα δύο τανκς είναι συμπαίκτες και προστατεύουν το κάστρο από npc αντιπάλους (sprites) που κάνουν spawn σε διάφορα σημεία του ταμπλό.

Σε κάθε περίπτωση υπάρχουν σαφείς συνθήκες νίκης και ήττας και καταμέτρηση health και score.

# ΙΔΙΑΙΤΕΡΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Κάτι πολύ ιδιαίτερο που αναπτύχθηκε σε αυτό το παιχνίδι είναι η οργανωμένη αρχιτεκτονική του.   
Πρώτον, όσον αφορά το διαχωρισμό και την κατηγριριοποίση των κλάσεων που περιγράφουν συγκεκριμένες λειτουργίες ή αντικείμενα (για παράδειγμα η κλάση tank). Δεύτερον, όσον αφορά το network connection, που αποτέλεσε και βασικό στόχο αυτού εδώ του project. Το μόνο που χρειάζεται είναι δύο διαφορετικές συσκευές υπολογιστών, συνδεδεμένες στο ίδιο δίκτυο Wifi.

Επιπρόσθετα, όπως φαίνεται και στις παρακάτω εικόνες, αξιοσημείωτο feature αποτελεί και το γεγονός ότι οι παίκτες μπορούν να κάνουν τόσο register, όσο και sign in, κάτι που υλοποιείται με τις κλάσεις που περιέχονται στο signInLevelSignUp.py του Client.

**ΕΝΟΤΗΤΑ 2η**

# ΤΕΧΝΙΚΕΣ ΛΕΠΤΟΜΕΡΕΙΕΣ

ΕΚΚΙΝΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

ΓΡΑΦΙΚΑ:

Σε πρώτο στάδιο, υλοποιήθηκε η κατασκευή απλών γραφικών και χρωμάτων, βασικών για τη λειτουργία της εφαρμογής. Αυτά τα γραφικά ήταν: το φόντο, τα μέρη των τανκς, το background image για το αρχικό παράθυρο, το κάστρο και οι σφαίρες.

Για την υλοποίηση του κώδικα χρησιμοποιήθηκαν εξ ολοκλήρου: η Pygame και το πρόγραμμα Microsoft Visual Studio Code, όπου εγκαταστάθηκαν οι αντίστοιχες επεκτάσεις, όπως το C# Dev Kit.

# ΑΡΧΙΤΕΚΤΟΝΙΚΗ

# CLIENT & SERVER

# ΕΠΕΞΗΓΗΣΗ ΤΟΥ ΚΩΔΙΚΑ & ΛΕΙΤΟΥΡΓΙΚΟΤΗΤΑ

Όπως φαίνεται και από το διαχωρισμό των αρχείων .py σε δύο φακέλους (Client Game και Server Game ), στην ουσία η εφαρμογή αποτελείται από δύο projects, τον client και τον server. Αν και αρκετές κλάσεις είναι κοινές και στα δύο, υπάρχουν κάποιες σημαντικές διαφορές που καλό είναι να τονιστούν.

* Η υλοποίηση βασικών αντικειμένων που βλέπει ο παίκτης στον υπολογιστή του και η γενικότερη δημιουργία του παραθύρου, γίνεται από τον Client. Δηλαδή, τα αντικείμενα δημιουργούνται τοπικά.

Φυσικά, έχουμε προβλέψει την αποστολή των δεδομένων από τον Client στον Server, αλλά εν γένει, τα στοιχεία δημιουργούνται στον Client και αποστέλλονται στον Server, με χρήση αρχείων json και βάσεων δεδομένων.

* Όλα τα δεδομένα, όπως όνομα username, password, ταυτόχρονη αντίληψη και εμφάνιση του αντιπάλου στέλνονται και αποθηκεύονται στον server.
* Για τη λειτουργία του παιχνιδιού, πρώτα ξεκινά να ‘’τρέχει’’ ο server και μετά ο client.
* Αν ο κάθε παίκτης συνδεθεί από διαφορετικό υπολογιστή, απαιτείται μόνο ένα run του project client game. Για να εμφανιστούν και τα δύο παράθυρα παικτών στον ίδιο υπολογιστή, όμως, απαιτείται ένα run και μετά ένα debug python file.

Ακολουθεί συνοπτική περιγραφή της λειτουργικότητας των διαφορετικών αρχείων, κλάσεων, μεθόδων και συναρτήσεων που περιέχονται στο server και στο client project.

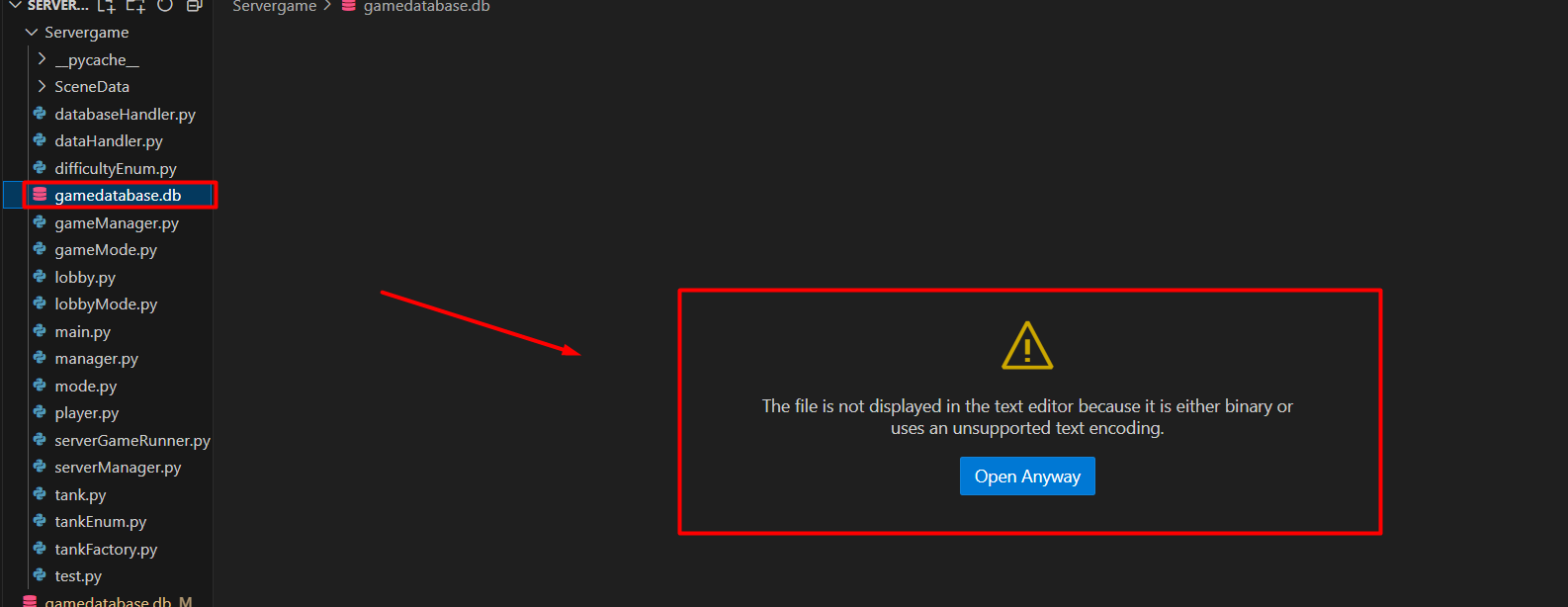
Α) SERVER

Η γενική λειτουργία του server είναι να λαμβάνει δεδομένα από τους χρήστες και να τα στέλνει πίσω. Τέτοια δεδομένα μπορεί να είναι, ποιο χρώμα tank διάλεξε ο παίκτης, που βρίσκεται ο αντίπαλος, η κατάσταση του health bar κλπ. Ο τρόπος με τον οποίο διατηρεί τα δεδομένα είναι με αρχεία json, φυλάσσοντας τα σε μία database.

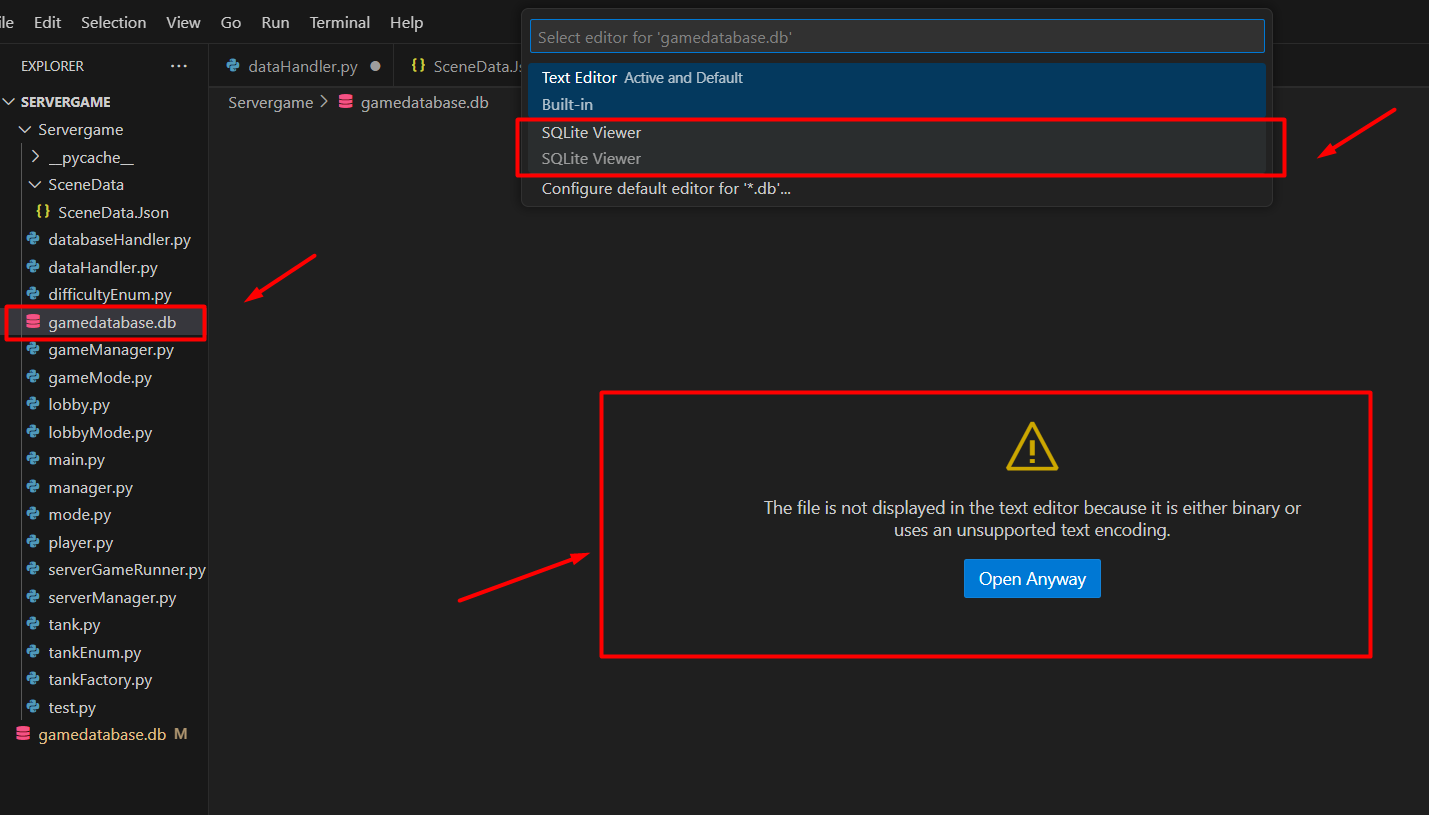
Ας εμβαθύνουμε στη λειτουργικότητα όλων των αρχείων (κυρίως κλάσεων) της μορφής .py που βρίσκονται στον Server.

Στην ουσία, όπως περιγράφεται και στο συνοδευτικό βίντεο, πρώτα ξεκινάει ο server και ανοίγει ένα παράθυρο που τρέχει το κεντρικό νήμα (thread) και δύο νήματα, ένα για κάθε παίκτη (νήμα container επεξεργασίας στον υπολογιστή). Αυτά τα δύο νήματα αναλαμβάνουν τη λήψη και τη μετάδοση δεδομένων σε σχέση με τον παίκτη. Πιο αναλυτικά, ακολουθεί περιγραφή όλων των λειτουργικών του αρχείων (κυρίως κλάσεων).

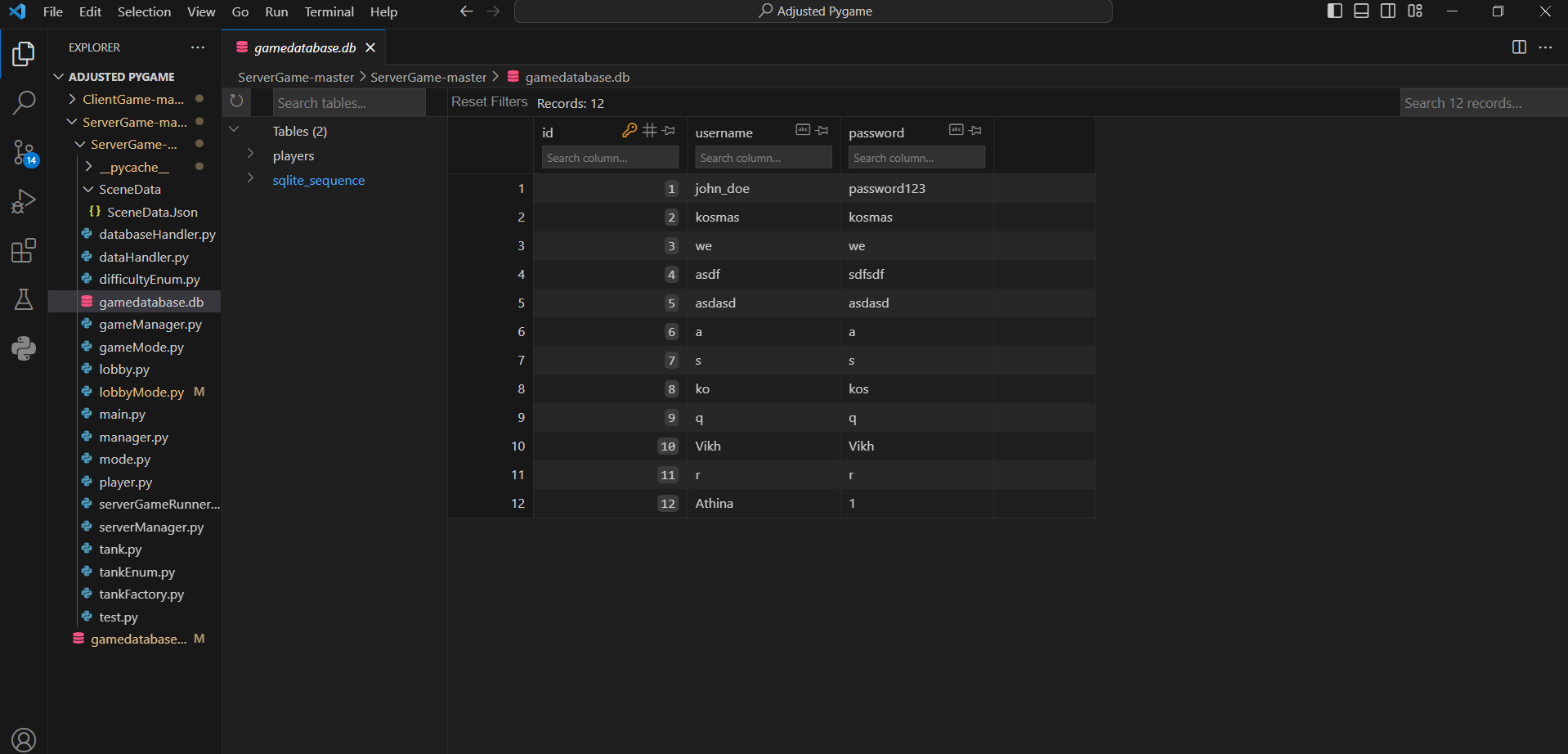
1. databaseHandler.py : εκτελεί δοσοληψίες ερωτημάτων (requests) στη βάση, στέλνοντας δεδομένα. Για παράδειγμα create, delet, κλπ
2. difficultyEnum.py : είναι στην ουσία μια κλάση αρίθμησης των επιπέδων του παιχνιδιού. Δηλαδή τα επίπεδα Easy, Medium και Hard, αντιστοιχίζονται στα value 1, 2, 3, αντίστοιχα. Ανάλογα με τη δυσκολία του παιχνιδιού, ο παίκτης έχει μικρότερο ή μεγαλύτερο fire rate και ζωή.
3. gamedatabase.db: είναι η βάση όπου φυλάσσονται όλα τα δεδομένα των παικτών. Διατηρεί όλα τα στοιχεία των παικτών, τα οποία αποστέλλονται (παράδειγμα για επαλήθευση) μέσω του databaseHandler. Αρχικά εμφανίζεται κρυμμένη αλλά μπορεί να φανερωθεί, αν ακολουθήσουμε τα παρακάτω βήματα, όπως περιγράφεται και στις εικόνες που ακολουθούν.



Αρχική εμφάνιση του database



1Άνοιγμα του database με SQLite Viewer



Εμφάνιση στοιχείων των παικτών

1. gameManager.py: αποτελεί μία βοηθητική κλάση η οποία, κατά τη διάρκεια του παιχνιδιού, συγκρατεί δεδομένα, τα οποία χρησιμοποιούνται σε διάφορες άλλες κλάσεις. Παραδείγματος χάριν, ποιος παίκτης αντιστοιχεί σε ποιο τανκ, ποιο είναι το username του παίκτη
2. se polles klaseis opws poio paixths exei pio tank me poio username poso firerate kai oposo zwh exei jo kathe paixths
3. poso zwh exei epishs kapoia instance ths klaseis paixtrhs h opoio krataei kapoia alla dedomena
4. gameMode.py :

>> to gamemode einai h klash pou poaizei to paixnidi dhladh ekei pou stelnontai data gia thn topothesia ton paixtwn tou kastrou to exthwn

to ana kapoios exei varesei me to kanoni to ana kapoios exei xasei zwh kai outw katheksh

1. lobby.py: λειτουργεί ως βοηθητική κλάση στο lobbyMode, η οποία διατηρεί τα δεδομένα αρχείων json, ώστε να μπορεί να τα αποστείλει το lobbyMode στους clients.
2. lobbyMode.py: συγχρονίζει τα δεδομένα στο lobby των παικτών. Δηλαδή, την επιλεγμένη δυσκολία, το επιλεγμένο τανκ και ο ready.
3. main :

>> h main einaiaa po ekei pou kszekinaei na trexei to programmat tou server tou paixnidiou

kai kaloi tiw alles ServerGameRunner

1. manager

>> h klash manager sketo einai mh hlopoihmenh skopos htane na afairei kai na prosthetei paiktes sto paixnidi px pethane o enas vgalton teleios apo to paixnidi

1. mode
2. >> mode einai h base kalsh gia ola ta mode krataei dedomena opws megethos parathurou xrwma data paixth opws einai to json des datap1
3. sthn periptwsh auths ths klashs auto to arxeio kanei anathsesh timhs sto runtime (otan ksekinas na pazizeis to paixnidi)
4. player

>> h klash player exei username vehicle an einai o prwtos h deuteros paixths pou sundeetai poso zwh exei kai to firerate

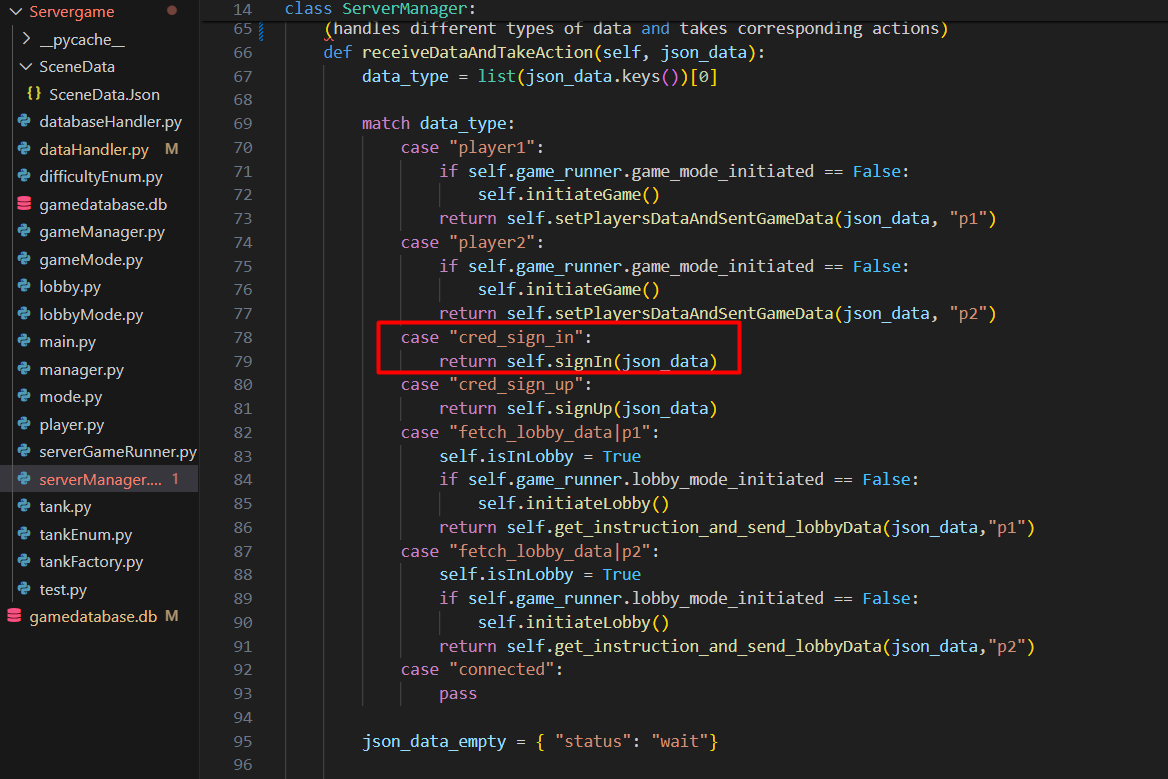
1. serverGameRunner

>> ServerGameRunner trexei to vasiko parathuro tou paixidiou einai einai upeuthiuo gia na parie kai na dwsei dedomena

opws einai ta events sthn periptwsh tou server den xrhsimopoiuntai kapou dioti den uparxei input apo to paikth

1. serverManager : είναι επί της ουσίας, η κλάση που αρχικά παραλαμβάνει τα δεδομένα των παικτών από τους εκάστοτε clients. Τα requests (πχ sign in, username, password) που γίνονται από τους Clients, στέλνονται εδώ, όπου γίνεται ένας έλεγχος επαλήθευσης του παίκτη και των δεδομένων του. Επίσης από εδώ εγκρίνεται η είσοδος του.

Αξιοσημείωτες αναφορές εντός του serverManager:

* Η init: παίρνει και αποθηκεύει την IP του Server, αρχικοποιεί τα flags των παικτών (‘self.p1\_in\_game’ και ‘self.p2\_in\_game’) και αρχικοποιεί ένα socket για server communication
* Η get\_public\_ip: χρησιμοποιεί ένα API (application programming interface) για να πάρει την public διεύθυνση IPv4.
* Η μέθοδος receiveDataAndTakeAction: αναζητά δεδομένα του τύπου player1, cred\_sign\_in, cred\_sign\_up, fetch\_lobby\_datalp1, connected, διχειρίζεται διαφορετικού τύπου data που στέλνονται από τους clients και, ανάλογα, εκτελεί τις αντίστοιχες εντολές. Στην ουσία, παίρνει το πρώτο node του json αρχείου και ανάλογα με την αντίστοιχη οδηγία, εκτελεί ένα action. Για παράδειγμα, ένα action pc sign in pairnei to usernbma ekai to

password kjaneio ena erwthma sthn meswq databnse handler kai epistrefei se mopfh jsonm analoga to ana exei paei kala to sign

in h oxi mhnhma paradeigma allo fethllobby pairni ta x y olwn twn ui entikeimenwn kai ta dinei stron client

opws epeish kai plhroforia gia to ana as pou h duskolia exie kleidwsei sthn peritwsh

pou h duskolia einai kleidwmenh stou clients emfanizetia me diaforetimo xrwma to ui th s duskolia easy hmediuy

hard

diafopretiko sprite allh zwh allh firetate klp

uparxei h methodos contrusct tanlk h opoia auto pou kanei einai analoga me to th dexetai os parametro se tank enum na

dhmiourtgei antikeimeno tetoio tupou

>> Sevrer manager base gia na pairnei kai stelnei json polu vasikh methodos

1. tank
2. >>tanks uparxoun 3 kloaseis sto arxeio tank.py h mia klash
3. einai h base kai exei kapoia dedomena genika gia to tank opws xy width kai heigh uparxei meta mia klkasei to tank base
4. to opio krataei thn eikona ths vvashs tou tank kai kinei tyo tank to tank turret einai mia kllash h opoia kinei to turret
5. kai to kanei rotate analoga me to pou einai to pontiki sthn othonh epeishs exei mia method firfe cannon h opoioa methodos
6. einai upoeuthunei gia na riksei vlhmata
7. epeish stis treis autes klaseis uparxei o sunxronsimos me to diktuo sthn ousia kataskeuazetai ena antikeimeno to opoio stelnetai sto diktuo
8. tankEnum.py : αποτελεί [enumeration](https://docs.python.org/3/library/enum.html) , το οποίο χρησιμοποιεί την κλάση Enum από το module enum της Python. Γενικά τα enumerations χρησιμεύουν στη δημιουργία named constant τιμών. Στη συγκεκριμένη κλάση, τα P1, P2, E1, E2 είναι τα 4 μέλη enumerators, ενώ οι τιμές 1, 2, 3,4 είναι οι αντίστοιχες τιμές που ανατέθηκαν σε αυτά (P1 = player1 = 1 , P2 = player1 = 2, E1 = enemy1 =3, E2 = enemy2 = 4). Στην ουσία εδώ γίνεται μια αντιπροσώπευση των τανκς, ώστε να έχουν νόημα σαν οντότητες εντός του κώδικα.
9. tankFactory.py: αποτελεί μία από τις πιο σημαντικές κλάσεις, η οποία περιέχει επίσης δύο κλάσεις, την init και την constructTank. Η κλάση tankFactory, ενσωματώνοντας το Factory Pattern, είναι υπεύθυνη για την κατασκευή αντικειμένων τύπου τανκς. Η χρήση της init εδώ είναι αυτή του constructor, που κάνει αρχικοποίηση των image paths, ώστε να δημιουργηθούν τα τανκς. Αξίζει να αναφερθεί ότι τα τανκς αποτελούνται από δύο μέρη, τη βάση, η οποία μετακινείται μπρος πίσω και το πάνω μέρος, το οποίο περιστρέφεται ώστε να βαράει προς κάθε κατεύθυνση.

u tasnk pc oi extroi exoun allaxarhskthtistk aopws

1. test.py: η κλάση αυτή ορίζει τη συνάρτηση recieveData, η οποία δέχεται ως παράμετρο μια δομή δεδομένων τύπου json\_data. Η συνάρτηση εξάγει τον τύπο δεδομένων από τα keys του dictionary στο json\_data Έπειτα, χρησιμοποιεί μια δήλωση match για να καθορίσει τον τύπο δεδομένων και να εκτελέσει συγκεκριμένες λειτουργίες, βασισμένες σε αυτόν τον τύπο. Ειδικότερα, η εντολή data\_type = list(json\_data.keys ())[0] , είναι αυτή που εξάγει το πρώτο key από το dictionary, θεωρώντας ότι αντιπροσωπεύει τον τύπο δεδομένων, ενώ με την παρακάτω δομή, ελέγχεται αν ο τύπος δεδομένων είναι cred\_sign\_in, cred\_sign\_up ή game\_data και ανάλογα εκτυπώνει αντίστοιχο μήνυμα ή όχι.

match data\_type:

case "cred\_sign\_in":

print(1)

case "cred\_sign\_up":

print(2)

case "game\_data":

pass

Τέλος με την εντολή receiveData(data\_to\_server), καλείται η συνάρτηση, δημιουργώντας μια προσομοίωση sign-in request και αντίστοιχα τυπώνοντας 1 για το cred\_sign\_in.

Β) CLIENT

Η γενική λειτουργία του client είναι να δημιουργεί δεδομένα, όπως είναι ο σχηματισμός του παραθύρου., η ανανέωση των frames κλπ. Επίσης τέτοια δεδομένα μπορεί να είναι ποιο χρώμα tank διάλεξε ο παίκτης, που βρίσκεται ο αντίπαλος, η κατάσταση του health bar κλπ. Ας εμβαθύνουμε στη λειτουργικότητα όλων των αρχείων της μορφής .py που βρίσκονται στον Client.

* Αν ο κάθε παίκτης συνδεθεί από διαφορετικό υπολογιστή, απαιτείται μόνο ένα run του project client game. Για να εμφανιστούν και τα δύο παράθυρα παικτών στον ίδιο υπολογιστή, όμως, απαιτείται ένα run και μετά ένα debug python file.
* Τα δεδομένα τυπικά καταχωρούνται μόνο στο Server, παρόλα αυτά κάποια προσωρινά δεδομένα που αφορούν την οπτική υλοποίηση

1. clientGameManager
2. clientNetworkManager
3. entryLevel
4. gameLevel
5. level.py
6. lobbyLevel.py
7. main.py
8. player.py
9. signInLevelSignUp.py
10. tank.py
11. tankEnum.py
12. tankFactory.py

>>clientGameManager: ελέγχει το level του παίκτη, υπεύθυνη για να αλλάξει το level

>>clientNetworkManager: δίνει και παίρνει πληροφορίες του παίκτη και τ απαει στο δικτυο, αυτή κάνει τη σύνδεση με το server

>>entrylevel: ειναι το 1ο level με το οποιο συνδεεται, εμφανιζει το παραθυρο, πατας το connect και σε συνδεει

>>Gamelevel: ειναι η κεντρικη πιστα του παιχνιδιου, αυτή ζωγραφιζει το παραθυρο, τους παικτες, το κάστρο, τα παντα...

και στελνει και παιρενι δεδομενα αναλογα με την τοποθεσια των παικτων

(εδω τροποποιουμε το μεγεθος παραθυρου, θεση αρχικη παικτη)

>> level: ιδιο περιπου με αυτην του server, είναι h base κλαση gia ta υπολοιπα level

>> levelManager: δεν υλοποιηθηκε

>> lobbylevel: η κλαση που εχει το παραθυρο που ζωγραφιζει το lobby οπου επιλεγεις επιπεδα κ ready

και στελνει τα αντιστοιχα δεδομενα στο server

>> main: τρεχει ολο το παιχνιδι από εδωιδια με του server παρομοια λειτουργια απλα γ τον client,

>> manager: half

>> signInLevelSignup: εχει 2 κλασεις- ειναι αρχειο- εδω σχηματιζεται το παραθυρο που κανεις

sign in, sign up και στελνει τα αντιστοιχα δεδομενα στο server

>> player: ιδια με του server, αρχιτεκτονικη τυπου client server & client authorotive

>>tank ομοια με server

>> tankFactory: dhmioyrgei to τανκ (bazv κ url)

# ΠΡΟΚΛΗΣΕΙΣ & BUGS

Κατά τη διάρκεια υλοποίησης του application, όπως και σε κάθε δημιουργική διαδικασία, εμφανίστηκαν κάποιες δυσκολίες, κυρίως όσον αφορά το συντονισμό Server και Client. Ειδικότερα, λόγω της αρχιτεκτονικής του παιχνιδιού, οποιαδήποτε μικρή αλλαγή στον κώδικα του ενός, σήμαινε αλλαγή και στον κώδικα του άλλου, με αποτέλεσμα να γίνει

**ΕΝΟΤΗΤΑ 3η**

# ΙΣΤΟΡΙΚΟ ΥΛΟΠΟΙΗΣΗΣ

Από την αρχική ημερομηνία ανάθεσης, το πρώτο βήμα ήταν η απόφαση του concept και των mechanics του παιχνιδιού. Ομολογουμένως, λόγω και μιας άγνοιας κινδύνων (υπό την έννοια του τι είναι εφικτό, υλοποιήσιμο, χρονοβόρο, σύνθετο ή όχι τόσο όσον αφορά την ίδια την Python και κατ’ επέκταση την Pygame), η έννοια του concept μας απασχόλησε ιδιαίτερα. Και αυτό επειδή, δεχτήκαμε την πρόκληση και των δώδεκα ζητούμενων της εργασίας. Των 7 ελάχιστων απαιτήσεων και των υπόλοιπων 5 επιπλέον επιθυμητών.

Παράλληλα με τον περιορισμένο χρόνο που είχαμε, ξεκινήσαμε από στοιχειώδη μελέτη της Python, του περιβάλλοντος εργασίας μας (Visual Studio Code), φτάνοντας μέχρι και πολύ προχωρημένη μελέτη, όσον αφορά τις βάσεις δεδομένων και την ικανότητα συνδεσιμότητας σε διαδίκτυο.

Κάθε κομμάτι του κώδικα, προέκυψε από trial and error, αμέτρητες αναζητήσεις στο google και το stackOverflow, αλλά και από επίσης άπειρους διαλόγους με το <https://chat.openai.com/> . Ιδιαίτερη χρησιμότητα είχε και στη διαδικασία του debugging. Μεγάλη βοήθεια στάθηκαν και κάποια video tutotrials, στην πλατφόρμα του youtube.

Βέβαια, η διαδικασία φιλτραρίσματος από τη γιγάντια υπερπληθώρα πληροφοριών, κάποιας χρηστικής οδηγίας, που πράγματι να λειτουργούσε στο περιβάλλον και τη δομή του παιχνιδιού, αποτέλεσε άθλο.

Κάθε φορά , όμως, που κάποιος συνδυασμός πληροφοριών με κάτι που είχαμε ήδη γράψει, λειτουργούσε, η χαρά μας ήταν απερίγραπτη. Αυτό ακριβώς μας ώθησε να συνεχίσουμε φτάνοντας το application σε ένα σχεδόν ολοκληρωμένο σημείο, παρέχοντας του χαρακτηριστικά και δυνατότητες επέκτασης και για περαιτέρω εξέλιξη.

# ΠΕΡΑΙΤΕΡΩ ΕΞΕΛΙΞΗ

Σχετικά με τις δυνατότητες βελτίωσης του project, μπορεί εύκολα να παρατηρήσει κάποιος, ότι καθ’ όλη την έκταση του κώδικα τόσο στο server όσο και στον client, υπάρχουν κάποια half implemented features (όπως η κλάση manager στο Server), τα οποία ενδεχομένως θα μπορούσαν να εμπλουτιστούν μελλοντικά

για την αναβάθμιση της εφαρμογής. Επίσης, χάρη στην οργανωμένη δομή του, τα σχόλια που εμπεριέχονται σε κάθε αρχείο και την αρχιτεκτονική του, μπορεί εύκολα να υποστεί επεξεργασία από τρίτους. Γενικότερα, σε ένα επίπεδο πιο προχωρημένο και ίσως και επαγγελματικό, υπάρχουν οι βάσεις για εξέλιξη του έργου. Άλλωστε, σε τέτοιου τύπου projects, συνήθως υπάρχει βελτίωση, όσο περισσότερο διάστημα ασχοληθούν οι δημιουργοί. Το έναυσμα δόθηκε. Από εκεί και πέρα, οι δυνατότητες είναι απεριόριστες, αν κάποιος απλά αφοσιωθεί στην υλοποίηση αυτή.

# ΣΥΜΠΕΡΑΣΜΑΤΑ

Ολοκληρώνοντας την παρουσίαση και περιγραφή του παιχνιδιού που υλοποιήσαμε, προκύπτουν κάποιες συμπερασματικές σκέψεις.

Η Pygame, αλλά και η ίδια η Python, αποτελεί ένα πολύτιμο εργαλείο για τον εν δυνάμει game developer, λόγω της ευκολίας που παρέχεται στην εκμάθηση και κατανόηση του κώδικα, των βιβλιοθηκών και των εντολών. Παρόλα αυτά, ενδείκνυται περισσότερο για 2D παιχνίδια. Παραμένει ωστόσο κατάλληλη για την ομαλή ένταξη αρχάριων στα βαθιά concepts του προγραμματισμού και συγκεκριμένα του game developing.

Κρίνοντας επίσης από την πολυπλοκότητα και το χρόνο που απαιτήθηκε για την αρχιτεκτονική Client Server, είναι ένα project που απαιτεί πλήρη ενασχόληση μαζί του, αποκλείοντας ενδεχομένως άλλες υποχρεώσεις.

Παρόλα αυτά, η ενασχόληση με το project αυτό, αποτέλεσε ένα καλό εναρκτήριο λάκτισμα, για τη βαθύτερη ενασχόληση με τον προγραμματισμό και κατανόηση βασικών εννοιών, ειδικότερα τον κλάδο κατασκευής παιχνιδιών. Σίγουρα έχουμε πολλά ακόμη να μάθουμε και, όπως αναμένεται με κάθε δημιουργική διαδικασία, τα λάθη, τα bugs, οι δυσκολίες, είναι κομμάτι της διαδρομής αυτής και όχι ανασταλτικός παράγοντας.

Παιδαγωγική σημασία

Εν κατακλείδι, ακόμη και αν το παιχνίδι έμεινε εν μέρει ελλιπές και κάποιοι αρχικοί στόχοι εν τέλει δεν έγιναν εφικτοί ακόμη (με τους κώδικες που αναφέρθηκαν πιο πάνω), τα ακόλουθα απαιτούμενα έχουν υλοποιηθεί ως εξής:

* Ύπαρξη τουλάχιστον ενός παίκτη, για την ακρίβεια δύο παικτών με διαφορετικά χαρακτηριστικά
* Ύπαρξη ταμπλό – πίστας, όπου βρίσκονται ένα κάστρο και 2 τανκς.
* Ύπαρξη τριών επιπέδων δυσκολίας (Easy, Medium, Hard)
* Αποθήκευση σκορ για τον παίκτη και «ανέβασμα» στον server
* Ύπαρξη γραφικών, όπως είναι τα τανκς που

# ΕΥΡΕΤΗΡΙΟ ΟΡΩΝ

Σε αυτό εδώ το παράρτημα, αναφέρονται εξειδικευμένες ορολογίες και έννοιες.

* Αρχείο [JSON](https://realpython.com/lessons/what-is-json/) (JavaScript Object Notation): είναι μία δημοφιλής μορφή αρχείου, που χρησιμοποιείται για την αποθήκευση προσωρινών δεδομένων, όπως δεδομένα ιστοσελίδων ή διαδικτυακών εφαρμογών. Αποτελεί συνήθης τακτική, η χρήση αρχείων json για τη μετάδοση και παραλαβή δεδομένων (σε μορφή κειμένου) μεταξύ ενός server και μιας δικτυακής εφαρμογής. Τα αντικείμενα σε ένα αρχείο json, αντιπροσωπεύονται ως ζευγάρια name/value. Με την εντολή *import json* στην Python, μπορούμε να αρχίσουμε να χρησιμοποιούμε αρχεία json.
* [Factory Pattern/Method](https://medium.com/@digit.sensitivee/game-programming-patterns-the-factory-method-87d7f8c12081): είναι ένα σχεδιαστικό πρότυπο, που χρησιμοποιείται κυρίως για τη δημιουργία αντικειμένων (objects), χωρίς να καθορίζουμε ακριβώς την κλάση του αντικειμένου που θα δημιουργηθεί. Συνήθως, δημιουργούμε μία γενική αφαιρετική κλάση, η οποία περιέχει τις μεθόδους που δημιουργούν αντικείμενα, συγκεκριμένες κλάσεις που την επεκτείνουν και αντιπροσωπεύουν τους τύπους των αντικειμένων, μια διεπαφή ή κλάση τύπου Factory που δηλώνει τη μέθοδο για τη δημιουργία αντικειμένων της διεπαφής και είναι υπεύθυνη για δημιουργία περιπτώσεων, συγκεκριμένες κλάσεις τύπου Factory που είναι υπεύθυνες για τη δημιουργία συγκεκριμένου τύπου αντικειμένου και ένα αρχείο τύπου client code, που καλεί τη μέθοδο factory για να δημιουργήσει objects instances, χωρίς να χρειάζεται να γνωρίζει μια συγκεκριμένη κλάση ή συγκεκριμένο αντικείμνενο.
* [Gameplay](https://en.wikipedia.org/wiki/Gameplay): η γενικότερη εμπειρία που βιώνει ένας παίκτης παίζοντας ένα παιχνίδι. Συνδέεται με στοιχεία όπως: τα mechanics του ίδιου του παιχνιδιού, τους κανόνες, τις προκλήσεις, τους στόχους, το flow και τη γενικότερη διαδικασία απορρόφησης και διάδρασης του παίκτη στον κόσμο του παιχνιδιού.
* [Sprite](https://en.wikipedia.org/wiki/Sprite_(computer_graphics)): 2D εικόνα ή animation που ενσωματώνεται σε μια σκηνή. Συχνά τα sprites χρησιμοποιούνται για να απεικονίσουν χαρακτήρες, παίκτες, npc, αντικείμενα και γενικά οποιοδήποτε είδους γραφικό.
* [Pygame](https://www.pygame.org/wiki/about): είναι ένα γκρουπ από modules της Python, το οποίο περιέχει ειδικές συναρτήσεις, μεταβλητές και κλάσεις, σχετικές με την ανάπτυξη παιχνιδιών χρησιμοποιώντας την προγραμματιστική γλώσσα Python
* [Socket](https://realpython.com/python-sockets/): αποτελεί δομικό στοιχείο για δικτυακές εφαρμογές, λειτουργώντας σαν κόμβος, στη διαδικασία αποστολής και παραλαβής δεδομένων. Δημιουργούνται σαν δίαυλοι επικοινωνίας, από τον client για να συνδεθεί στον server και από τον server σε αναμονή σύνδεσης του client. Από τη στιγμή που οριστικοποιηθεί μια επικοινωνία μεταξύ client και server, ανταλλάσσονται ελεύθερα δεδομένα μεταξύ των αντίστοιχων sockets τους.