



# 4. Semesters Afsluttende Projekt

4A

Oliver Boots

02. Juni 2024

Københavns Erhvervsakademi

Vejledere:

Kevin Lindemark Holm

Anslag med

mellemrum:

47.416

## 1. Resume

Denne rapport tager udgangspunkt i de virtuelle markeder man kan finde i onlinespil, og undersøger mulighederne for en digital løsning til Final Fantasy XIV der kan tilføje værdi til spillernes interaktioner med disse markeder. Via en hjemmeside med brug af API'er fra XIVAPI og Universalis, kan brugeren søge på objekter på markedet og få overskuelige informationer omkring dens salgsværdi, samt hvilket komponenter der bruges til at vide præcist om der er mulighed for profit. Derudover kan brugeren søge på en spiller-karakter og få en kollektions-oversigt ud fra deres profil der er "scrapet" fra Lodestone, som er Final Fantasy XIV's officielle hjemmeside.

Programmet viser potentielle og det er muligt, hvis kortene bliver spillet rigtigt, at skabe profit gennem dette produkt via reklamer, donationer eller lave nye funktioner som brugere ønsker at betale for.

Siden kan tilgås her: <https://erunoma.pythonanywhere.com/>

**Log in info:**

**Username:** Erunoma

**Password:** KEA

## 2. Indholdsfortegnelse

<b>1. Resume</b>	<b>2</b>
<b>2. Indholdsfortegnelse</b>	<b>3</b>
<b>3. Indledning</b>	<b>4</b>
<b>4. Indledende Undersøgelse</b>	<b>5</b>
4.1 Introduktion	5
4.2 Kvantitativ undersøgelse - Brugerundersøgelse	10
<b>5. Projektbeskrivelse</b>	<b>13</b>
<b>6. Ideudvikling</b>	<b>15</b>
6.1 Persona & Userstories	15
<b>7. Kravspecifikation</b>	<b>17</b>
<b>8. Løsningdesign</b>	<b>19</b>
8.1 Kodebeskrivelse	23
8.2 Visuel Design	32
<b>9. Test af løsning</b>	<b>32</b>
<b>10. Implementering af løsning i drift</b>	<b>35</b>
<b>11. Praktisk projektplanlægning og ledelse</b>	<b>36</b>
<b>12. Konklusion</b>	<b>39</b>
<b>13. Projektforløb</b>	<b>40</b>
<b>14. Perspektivering</b>	<b>41</b>
<b>15. Litteraturliste</b>	<b>42</b>
<b>16. Bilag</b>	<b>43</b>

### 3. Indledning

I vores verden vi lever i, er størstedelen af produktionen blevet organiseret gennem “supply-chains” og vægtet mod “Supply & Demand”.

Sælgere ønsker at få største udbytte af deres investering, mens køberen ønsker at betale så lidt som muligt for varen. Det har derfor været i alles interesser at udvikle værktøjer til at hjælpe med at få højest udbytte af en handel. Hvis vi ser det fra forbrugerens side, findes der hjemmesider såsom Pricerunner, der finder sider hvor man kan købe en vare for den billigste pris. Samtidig prøver sælgeren at konkurrerer med andre sælgere, for at få deres vare solgt.

I online spil med virtuelle penge, forholder spillerne sig til penge typisk på en anden måde end den virkelige verden. Der er større chance for at penge bliver brugt frugalt eller uden omtanke, men der er ligeså chancen for at nogle gør det til et mål, at anskaffe sig så mange penge som muligt.

Spillernes forhold til den virtuelle valuta beholder sig uden for den fysiske verden, indtil man kan direkte købe valuta med en rigtig valuta såsom den danske krone.

Historien bag virtuelle markedspladser i spil går flere årtier tilbage, men i scopet af denne rapport, beholder vi omtalen indenfor en vis kategori af online spil; Massive Multiplayer-Online spil, også kaldet MMO’s.

Det interessante ved MMO’er, er grænsen mellem et spil for underholdning, og en verden som er befolket af mennesker, som samles i et unikt form for fælleskab(Rapp A., 2018) Både den virkelige og den virtuelle verden påvirker hindanden begge veje, og det er interaktionen mellem spillets mekanikker, markedspladsen og spillets design der er særlig unikt.

Når der er andre mennesker omkring én, er den sociale status en vigtig del af spillerens interaktioner med resten af spillerne. Den status kan anskaffes på adskillige måder. Om det er ud fra venlighed/hjælpsomhed, ens egenskaber i kontekst af spillet eller hvorvidt man kan vise sig frem med de dyreste objekter i spillet. Da MMO’er går efter at folk socialisere, er anskaffning status en naturlig efterfølger.

## 4. Indledende Undersøgelse

### 4.1 Introduktion

Inden der kan søges efter et potentiel problem, skal vi først have defineret nogle key-pointers, som f.eks. hvad et MMO egentlig er og hvordan spillere integrerer med markedet i de diverse spil.

Som tidligere defineret i indledningen, står MMO for Massiv-Multiplayer-Online. Definitionen er lavet på baggrund af spillets infrastruktur, hvor spillere fra hele verden kan logge ind, typisk med en fiktiv karakter og se alle andre spillere der er i samme miljø.

Det er muligt at samle flere hundrede spillere på samme sted og der er blevet lavet arrangementer og events der bliver holdt inde i disse virtuelle verdner.

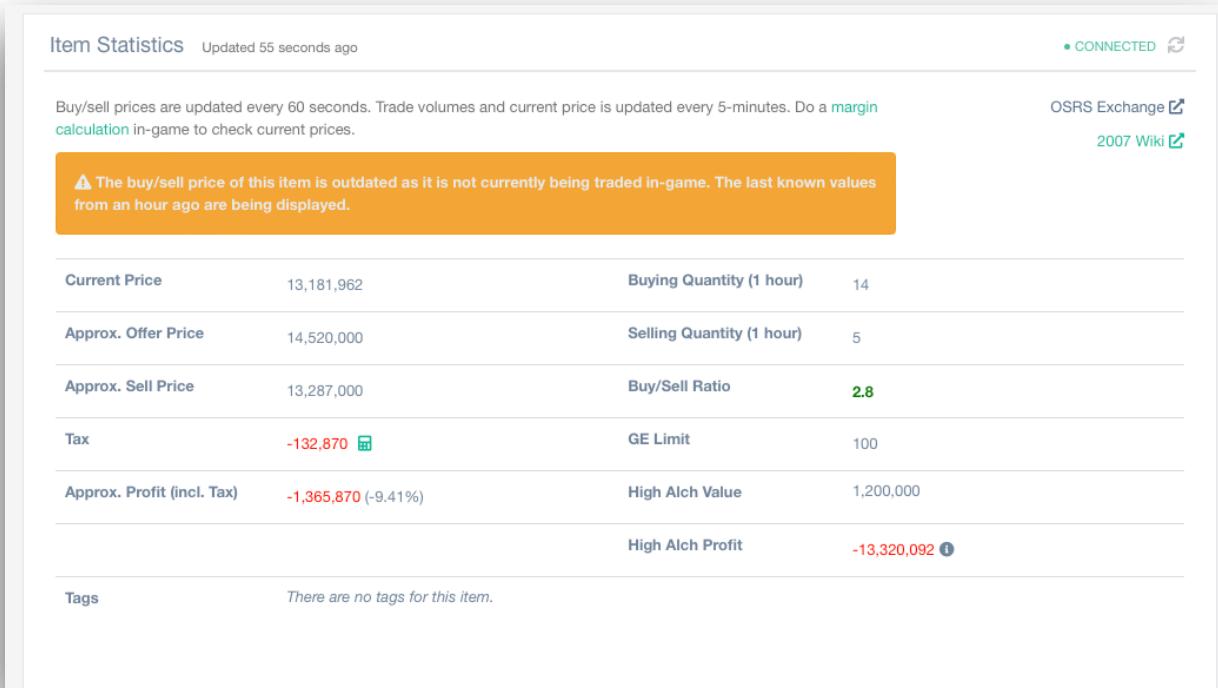
Afhængig af spillet, er penge vigtige for at kunne anskaffe sig materialer til at fremføre ens fremgang gennem spillet, og en god del af spillere bruger markedspladsen til at anskaffe sig enten materialer eller sælge det de har i deres besidelse til at få kapital til andre investeringer.

Der findes hjemmesider såsom Universalis eller GE Tracker, der lader brugeren se statistikker på prisen af et objekt eller materialer. Begge eksempel-hjemmesider bruges til FFXIV og Runescape hvert i sær.

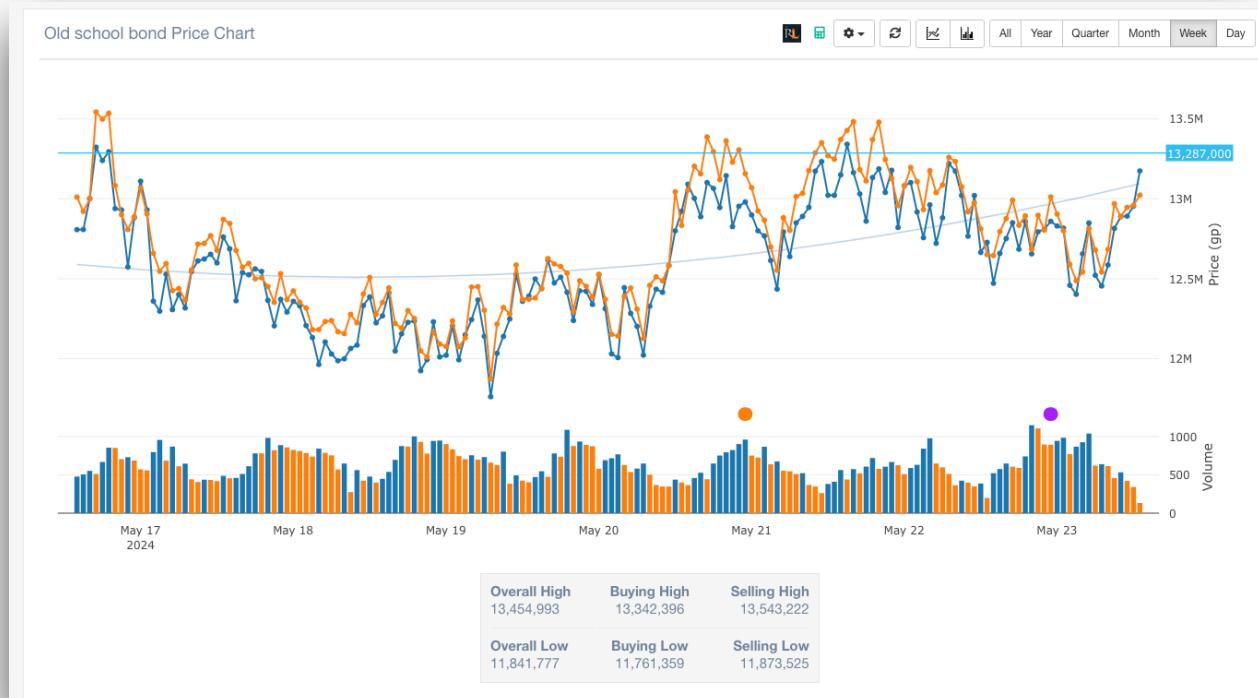
Hvert spil har deres egen unikke måde deres marked fungere på, med massere af nuancer og markedsteorier der kan bruges.

#### 4.1.1 GE-Tracker

GE-Tracker er en hjemmeside med detaljeret rapporter og interaktive værktøjer der kan hjælpe den enkelte bruger<sup>1</sup> med at få overblik over markedet og hvad der aktuelt bliver solgt, med grafer og tabeller. Derudover er der adgang til andre funktioner såsom “Decant Potions” og visse andre funktioner der er betalt månedligt.



[Figur 1 - GE-Tracker.com]



[Figur 2- GE-Tracker.com]

<sup>1</sup> Brugere og spillere er set som synonymer og er essentiel samme gruppe. Brugere er dem som “bruger” et værktøj og spillere af dem som “spiller” spillet

## 4.1.2 Universalis

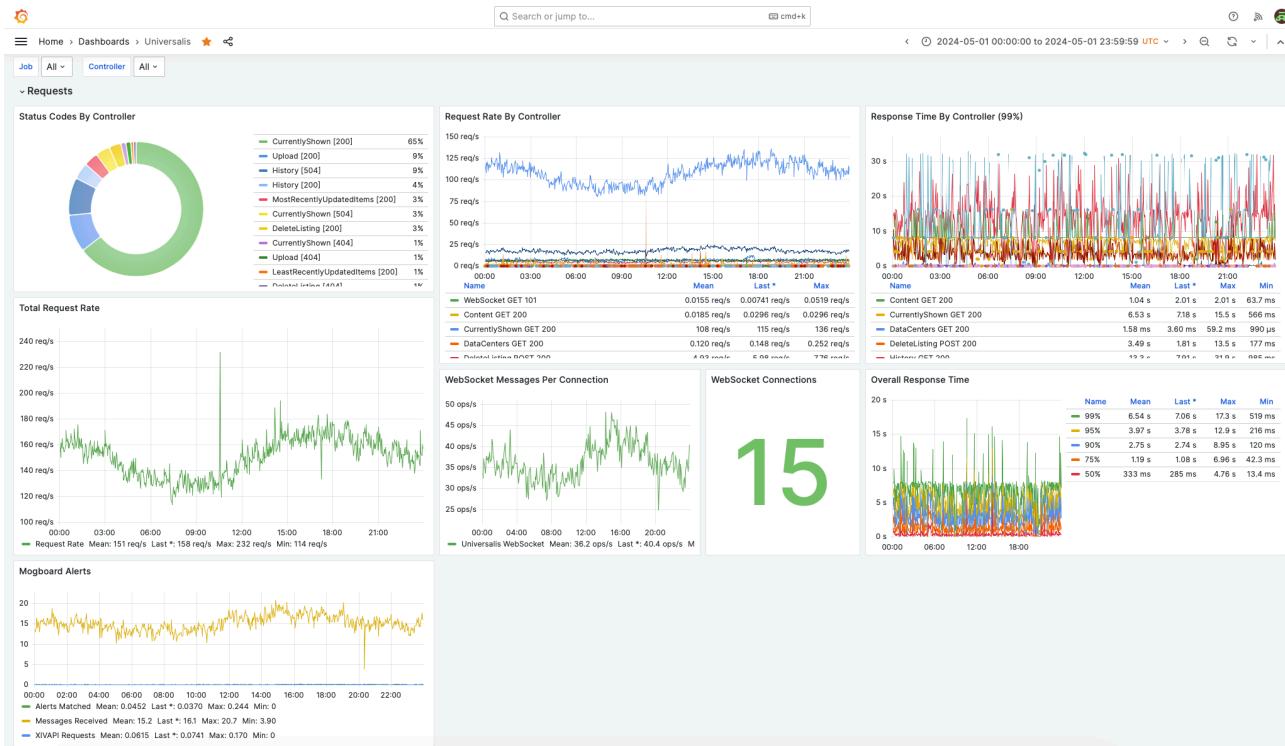
Universalis er en crowdsourced aggregator der indsamler oplysninger omkring markedet fra doneret information via deres selv-udviklet API.

Deres opsætning af det individuelle objekt ser på hver server og sammenligner, i en liste sorteret fra billigst til dyreste, hvor man kan finde dem henne.

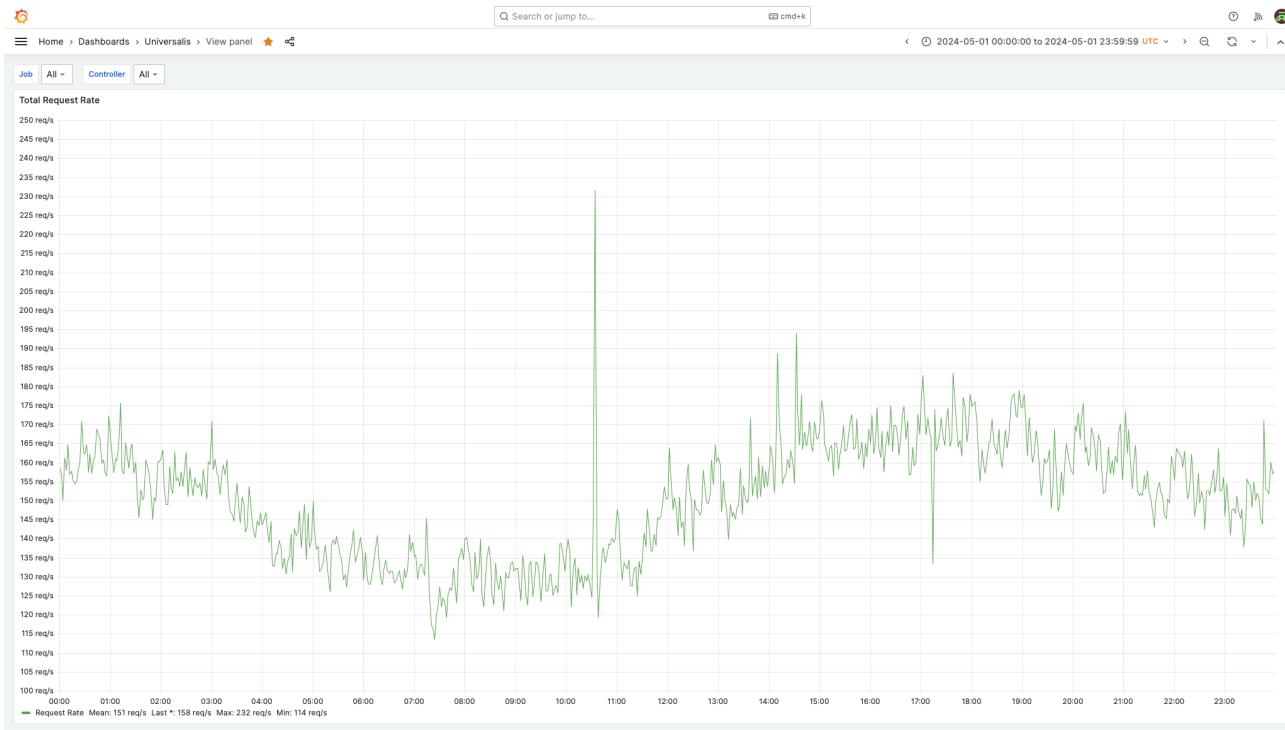
The screenshot shows the Universalis app interface for the item "545 Chondrite Magitek Samurai Blade". At the top, there's a search bar with "chondrite" and a "Market" tab. On the right, there are links for "Login via Discord" and a gear icon. Below the header, the item name is displayed with its ID, a thumbnail image, and a note: " Arms - Samurai's Arm - Stack: 1 - 88 SAM". To the right are buttons for "Saddlebag Exchange", "GarlandTools", and "Teamcraft". The main content area has tabs for different servers: Europe, Chaos, Light, Alpha, Lich, Odin, Phoenix, Raiden, Shiva, Twintania, and Zodiark. Below these are tabs for Oceania, Materia, Bismarck, Ravana, Sephirot, Sophia, and Zurvan. Under each server tab, there are sections for "ALPHA", "LICH", "ODIN", "PHOENIX", "RAIDEN", "SHIVA", "TWINTANIA", and "ZODIARK", showing the price and time since update. A "CHEAPEST HQ" section shows a price of "1 x 45,000" with a note "Server: Odin - Total: 45,000". A "CHEAPEST NQ" section notes "No NQ for sale.". Below these are tables for "HQ PRICES" and "HQ PURCHASE HISTORY", both showing transaction details like server, price, quantity, total, percentage change, buyer, and date. At the bottom, there are sections for "NQ PRICES" (no listings) and "NQ PURCHASE HISTORY" (no recorded history).

[Figur 3 - Universalis.app]

Ud fra Grafana statistikkerne af Universalis, målt over hele dagen d. 1 maj, kan man se en rate på omkring 100 - 200 requests per sekund, som går op og ned afhængig af tidspunktet på dagen. Dette viser en god del aktivitet på deres API.



[Figur 4 - Universalis Grafana Overview]



[Figur 5 - Universalis Grafana Requests]

Disse sider er lavet til formål at indsamle og vise markeds-data på en måde som ikke er indbygget i spillet selv. Afhængig af hvor ofte man ønsker at bruge disse værktøjer, kan effekten være enorm.

Der har været stor diskussion blandt spillere, hvorvidt disse værktøjer burde være acceptabelt at bruge.  
(Youtube, Reddit osv.)

Det hele ligger i en grå-zone, hvor det er svært at gennemskue hvad der er acceptabelt instinktivt og/eller lovligt. (Plektonlabs, 2021)

Ifølge Victoria McArthur i journalen: “Knowing, not doing/ modalities of gameplay in World of Warcraft”, er addons/plugins defineret, som redskaber der er til rådighed. Og hun tilføjer selv med at visse addons supplere og udvider spillerens kognitive evner i spillende. (McArthur, 2012)

Derudover, i adskillige diskussioner på online forums såsom Reddit, siges det generelt i fælleskabet, at alt der integrerer med spillet, både direkte og indirekte, som ikke er lavet af firmaet bag spillet selv, som tredje-partis-værktøjer<sup>2</sup> (TPT). Addons og plugins går også under denne definition.

Square Enix, som er udvikleren bag FFXIV, har fastlåst at ALLE TPT'er er forbudt. (The Lodestone, 2022)

Så for at bedre forstå hvad der ses som en TPT, og for at få et bedre overblik over hvilket problemstillinger spillere har i forhold til markedet, havde jeg lavet en brugerundersøgelse, som var sendt ud til forskellige fælleskaber der samler sig omkring FFXIV. Alle besvaret anonymt.

---

<sup>2</sup> Oversat fra det engelske ord Third Party Tools

## 4.2 Kvantitativ undersøgelse - Brugerundersøgelse

Brugerundersøgelsen kunne beskrives som en forundersøgelse, hvor spørgsmålende hjælper med at give et større indblik ind til typen af spillere og hvor positive de er omkring visse idéer. De havde også muligheden for at tilføje deres egne ønsker, som både gav mig inspiration, men også gav et tydeligere indblik i hvad potentielle problemer der ønskes løst.

Brugerundersøgelsen var skrevet på engelsk og fokusere på det engelsk-talende del af fælleskabet. Derfor får jeg desværre ikke svar fra spillere fra f.eks. Japan, som har en stor spiller-base i sammenligningen med det vestlige marked.

Undersøgelsen var delt op i to sektioner. Den ene fokuserede på at indsamle informationer omkring hvilket typer af spillere der er med i undersøgelsen. Deriblandt var spørgsmål omkring timer brugt om ugen på at være logget ind, hvilket aktiviteter de interessere sig indenfor og hvordan de føler personligt omkring TPT'er.

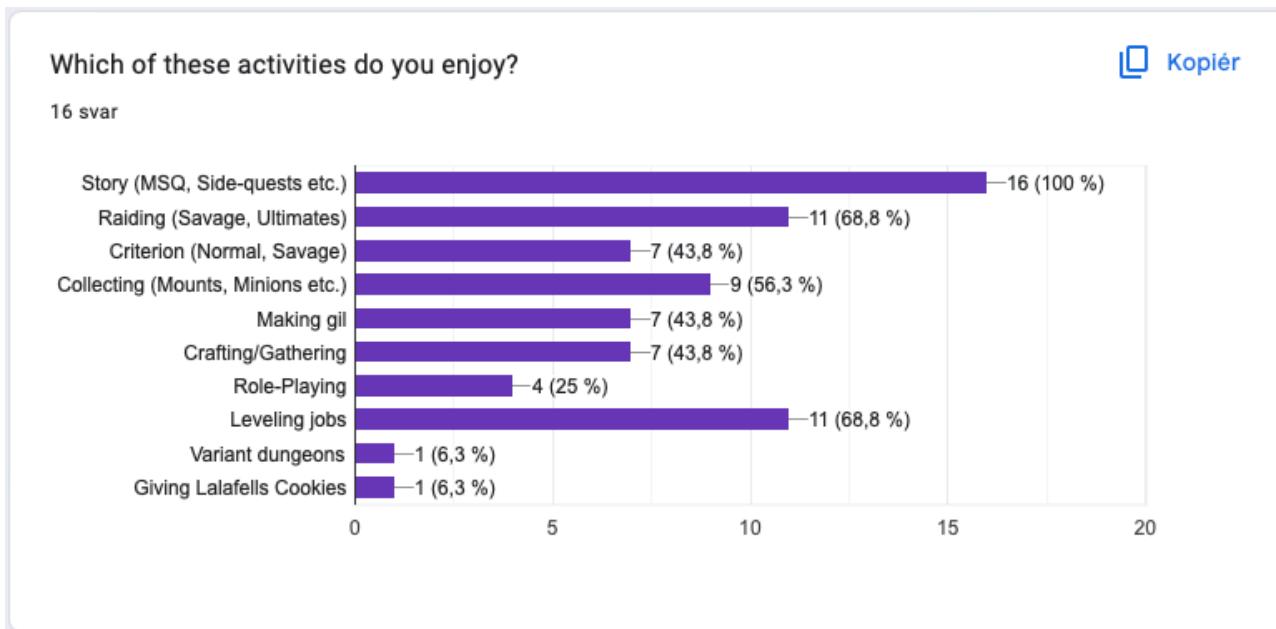
Den anden sektion fokuserede på at opstille scenarier og ideer til værktøjer og indsamle effektiviteten og entusiasme-niveau overfor disse værktøjer. Det var også muligt for brugeren at tilføje deres egne ideer som inspiration.

Det er intentionen med spørgeskemaet at indsnævre mulighederne, så der udvikles de funktioner der har størst gavn for de fleste spillere.

---

### 4.2.1 Svar og udbytte

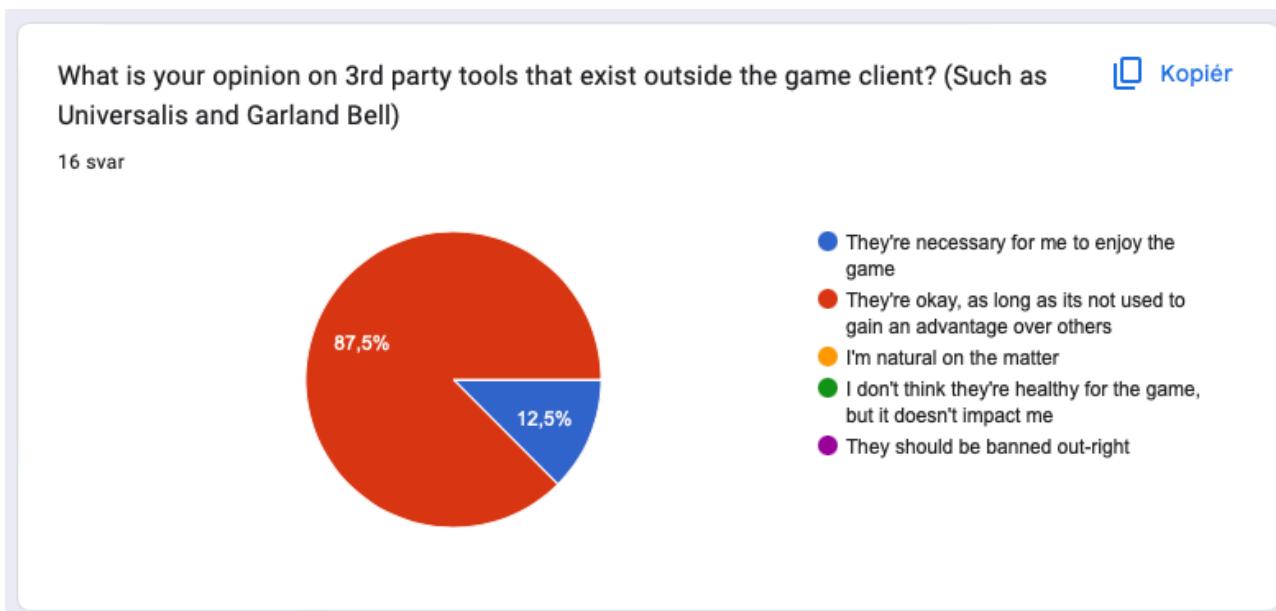
Ud fra resultaterne af 16 deltagere og som ses på figur 4, svarede 7 med at de godt kan lide at lave penge (Gil) i spillet. 9 Kan også lide at indsamle samleobjekter såsom mounts, minions og andet.



[Figur 6 - Brugerundersøgelse]

80% af deltagere har svaret i spørgsmålet hvad deres holdning er i forhold til TPT'er, at det generelt er okay, så længe det ikke giver en eksklusiv fordel.

Dette resultat er interessant fordi definitionen af "fordel" kan være ret subjektiv afhængig af personen.



[Figur 7 - Brugerundersøgelse]

I den 2. Sektion af formen, havde jeg opsat nogle scenarier for mulige værktøjer, og brugeren skulle give en score fra 1 - 5 om de kunne se dem selv bruge sådan et værktøj.

Der var to scenarier der er især vækker interesse hos brugerene:

Den første af at have et værktøj der lader én vælge samleobjekter de ikke allerede ejer, og giver dem en oversigt over det billigste opslag af de samleobjekter på markedet og giver dem en estimeret Gil-mængde de skal regne med at komme afsted med.

Den anden er et værktøj der lader brugeren vælge et objekt de kan lave, og få en oversigt over alle materialer der skal bruges til at lave dette objekt, samt en estimeret Gil-mængde de skal bruge, hvis de køber alle materialer fra markedet.

Grunden til at der er mest interesse for specifikt disse to eksempler, er at mange objekter (våben, rustning, samleobjekter osv.) kan sælges tilbage til andre spillere og det er her der er stor mulighed for profit. Et ofte-lavet trick er at købe lavt og sælge højt, et ganske enkelt koncept der bruges også i rigtige forhandlinger.



[Figur 8 - Brugerundersøgelse - Sektion 2]

#### 4.2.2 Spørgeskema-Resultat

Ud fra spørgeskemaet, indsat jeg at ikke alle går aktivt op i at lave Gil, men størstedelen laver aktiviteter såsom Raiding eller søger efter samleobjekter, som kan blive dyrt, hvis man køber det fra markedet. Derfor er der altid en vis tilskyndelse til altid at have Gil på hånden.

Der er en del der tyder på, at der ønskes features der gør det nemt at indsamle informationer der ville ellers være tidskrævende. Så data der kan samles hurtigt og opstilles pænt og gennemskueligt, har højest prioritet.

Det har desværre ikke været muligt at anskaffe en stor pulje af svar og det reflekteres i resultatet. En større del af svarende kommer fra den type af spiller der er dybt integreret i fælleskabet og bruger meget af deres tid på at interagere med spillet både online og offline, som kan dreje svarende i en bestemt retning.

Ud fra den indledende undersøgelse, blev projektbeskrivelsen lavet.

## 5. Projektbeskrivelse

Til dette projekt, laver jeg en hjemmeside der kan tilgås via nettet, som har til formål at levere en række forskellige værktøjer der kan hjælpe brugeren i at hurtigt finde informationer og holde styr på dele af deres interaktion med spillet Final Fantasy XIV.

Spillet kører på servere, som hver spiller er tildelt, eller kan vælge når de starter spillet. Hver server har deres egen markedsplads, hvor folk kan købe og sælge de varer de har anskaffet i spillet. Det er muligt at skifte hvilket server man er på, og mange bruger det til at spare nogle virtuelle penge.

Ved hovedsagelig brug af API'en fra siden: "Universalis", kan man indhente data fra spillet, og bruge det til f.eks. at opsætte værktøjer med. De værktøjer som laves under dette projekt, er som følger:

- At indhente karakter-data og opstille det, så brugeren kan nemt se hvilket ting i deres samling de mangler, og hvor billigst de kan købe det henne.
  - At indsamle og opstille et vindue med alle materialer og deres værdi, der er brug for til at lave et objekt i spillet. Objektet der ønsket laves kan søges efter.
  - At give en notifikation til brugeren, når et objekt ændrer sig heftigt i deres pris på det åbne marked.
- Notifikationen kan enten være lyd- eller visuelt repræsenteret.
- At opstille et vindue, der lader brugeren holde styr på alle materialer nødvendigt usædvanlige objekter og hvor man kan skaffe disse materialer udenfor markedspladsen.

Inspiration til disse værktøjer blev indhentet via en indledende undersøgelse foretaget med spørgeskema inden projektstart.

Løsningen laves med brug af Python og Flask og involverer brug af data-udvinding, samt data-opsætning via API'er, JSON og CSV. Dataen er visuelt repræsenteret på en brugervenlig måde via en HTML side der gør brug af Jinja/Javascript.

### Teknisk opsætning mm:

Programmet udvikles og testes via Visual Studio Code på en 2019 Macbook Pro.

Når hjemmesiden er klar til offentlig udgivelse, ville det være hostet gennem en hosting-side såsom Pythonanywhere eller med en lokal løsning såsom Raspberry PI.

---

## Afgrænsning af scope

Ud fra tidsgrænsen af projektet og de færdigheder jeg har på et 1-mands-projekt som dette, måtte der begrænses hvilket funktioner der sættes i prioritet og hvordan arbejdsmængden kan passe ind i tidsrammen. Hvis der var mere tid til projektet, eller hvis der kun var fokus på produktet selv, ville der kunne være plads til ekstra tilføjelser og endnu mere finpudsning af produktet.

Ud fra dette, ville der være funktioner der er med vilje frataget fra den original projektbeskrivelse, mens de vigtigste funktioner fokuseres på. Se Kravspecifikationen for et detaljeret overblik.

## 6. Ideudvikling

For at appellere til dem som ikke ofte aktivt prøver at anskaffe sig penge(Gil) og til dem som samler på samleobjekter, skulle løsningen være simpelt og lige-til, der giver de oplysninger der er brug for, uden at drukne brugeren i informationer de ikke nemt kan gennemskue.

Derudover skal der forhindres frustrationer ved brug af løsningen og lade brugeren føle sig komfortabelt og sikker med programmet. Derfor er en hjemmeside et åbenlyst forslag. Der kunne også bruges et program som downloades til PC'en eller lave det som en native-app til telefoner.

Men ud fra mine egne færdigheder og i interesse af instituttet denne rapport er skrevet under, ses en hjemmeside bygget på Flask med Python, som den bedste vej frem.

### 6.1 Persona & Userstories

I et spil så stort som Final Fantasy XIV med over 30 millioner registreret kontoirer (Twitch.tv, 2024) og som beskrevet i den indledende brugerundersøgelse, findes der flere typer af spillere med forskellige mål, ønsker og opførsler. Derfor udpeger jeg nogle særlige målgrupper via personaer og userstories herunder:

Jeg er en dreng der går på gymnasiet og startede med at spille efter jeg hørte omkring det fra en ven. Vi hænger ud og gør ting sammen når jeg endelig har tid, men når kun at være online et par gange om ugen. Det har ikke været muligt for mig at tage tiden til at anskaffe mig Gil, så jeg kan købe alt det rustning og våben jeg ønsker. Jeg føler mig ikke stærk nok og dør ret ofte, og jeg kender ikke til de effektive måder at anskaffe bedre "gear" på. Det ville være fedt hvis jeg kunne slå op visse våben osv. og finde en rimelig pris på dem, uden at skulle flytte over til de andre servere jeg har adgang til.	Story 1.1 Casual
Det kan godt være at jeg ikke er god til at kæmpe, men dér hvor min passion ligger, er at samle minions, mounts, og alt andet mellem himmel og jord. Jeg prøver alle slags aktiviteter, hvis der er noget man kan samle på. Min kollektion har dog en masse huller, og kan se at de fleste kan købes fra markedet. De er typisk til rimelige priser, men der kan være store forskelle i salgsprisen mellem servere. Men det er så forfærdeligt at manuelt skifte imellem dem, så ville ønske at der var en måde at tjekke priserne på alle serverene nemt og hurtigt, så jeg kan købe dem for den billigste pris.	Story 2.1 Collector
Når det gælder at indrette et hus i spillet, så er jeg én af de bedste. Jeg bliver endda betalt i Gil af andre spillere til at lave deres indretning for dem, og imellem den tid jeg bruger på mine klienter, indretter jeg mine egne boliger med designs jeg har lavet. Det er dog bekosteligt at anskaffe sig alle møbler, tapeter osv., og priserne på markedet er typisk tårnhøje. Det ville være rart hvis jeg kunne vide hvilket møbler der er hurtigst bare at købe på markedet, og hvad jeg kan spare mange penge på ved at lave det selv.	Story 3.1 Housing-Enthusiast

<p>At lave ting er hvad jeg bruger min tid på i FFXIV. Har mestret alle de forskellige færdigheder der findes og kan lave næsten alt. Afhængig af hvad der laves, kan jeg profitere, selvom der ikke er stor efterspørgsel. Det handler om at være på det rigtige sted og på det rigtige tidspunkt. Jeg ville ønske at der var en måde at se trenden i markedet.</p>	Story 4.1 Crafter
<p>For mig, handler det om at være der for mit team. Jeg forbereder mig til den grad som forventes, med at have potions og mad til at få det største boost til min performance. Har dog ikke tid til at lave mine egne, så bliver nød til at bruge dyre domme i at købe dem fra markedet. Det ville være rart ikke at miste så mange gil...</p>	Story 5.1 Raider

## 7. Kravspecifikation

Med udgangspunkt i at løsningen af lavet til dem der aktivt går efter at anskaffe sig Gil, og er investeret i spillet på et semi-dybt niveau, er mine kravspecifikationer som følger:

ID	Krav	Prioritet	Acceptest	Status
1	Brugeren kan via hjemmesiden søge efter et samleobjekt og får et nummer tilbage af den billigste pris der findes på markedet.	1	Hvis brugeren kan søge på navnet af objektet og får en akkurat tilbagemelding, ses kravet som bestået.	Passed/ Failed
2	Brugeren kan via hjemmesiden søge efter et objekt der kan sælges, og få oplysninger omkring materialerne der bruges til at lave det, og den estimeret pris, hvis materialerne skulles købes via markedspladsen.	1	Hvis brugeren får en detaljeret rapport tilbage på alle materialerne der bruges, og få et tæt estimeret pris på varende individuelt og tilsammen, ses dette krav som bestået.	Passed/ Failed
3	Hjemmesiden skal være beskyttet mod misbrug af API'en	1	Hvis der er sat foranstaltninger op til at formindske unødvendig brug af server ressourcer, både på backend og REST API'en, ses dette krav som opfyldt.	Passed/ Failed/ Tested
4	Hjemmesiden skal kunne tilgås fra internettet	1	Hvis man kan tilgå siden fra eksterne netværk, ses dette krav som opfyldt.	Passed/ Failed
5	Hjemmesiden skal kunne bruges både i mobil-format og desktop-format	2	Hvis hjemmesiden er tilgængeligt og kan bruges effektivt i flere skærm-formater, ses dette krav som opfyldt.	Passed/ Failed
6	Serveren er hostet på en Raspberry PI eller via en hosting side	2	Hvis serveren køres fra en enhed der kan køre 24/7, ses dette krav som opfyldt	Passed/ Failed
7	Brugeren kan slå op deres karakter og få en oversigt over deres samling. I deres samling kan de klikke på et objekt og få informationer omkring markedet om dette objekt.	2	Hvis en bruger kan se en karakter fra FFXIV visuelt og se deres samling, og få markeds-information omkring objekterne i samlingen, ses dette krav som opfyldt.	Passed/ Failed
8	Hjemmesiden skal være responsivt og virke professionelt	3	Hvis der ikke er lange vendte-tider uden visuelt feedback, ses dette krav som opfyldt	Passed/ Failed/ Tested

### Rationale for prioritering af krav

Ved projektets start og indtil kravene blev fastlagt, var det ikke klart hvor den største del af arbejdsbyrden lå henad. Ud fra det, gennemgik jeg forundersøgelsen for at bedre finde den korrekte prioritering af kraverne som nu er i kravspecifikationen.

Prioritering er oplagt i tre stadier:

Første prioritets opgaver ses som essentielle for MVP'en, og skal derfor være 100% funktionel inden deadline. Dette niveau af prioritering ses også som det potentielst mest brugbare for brugeren.

Anden prioritet sætter fokus på store features der kan have en større påvirkning på produktets endelige kvalitet, men laves efter det mest essentielle er udført.

Den tredje og sidste prioritet klassificeres som “rart-at-have”, det ville sige at de hjælpe med at gøre produktet mere fuldendt, men er ikke en del af kerne-funktionerne.

Fokuset med dette projekt er at have adgang til siden fra internettet.

I de tidligere projekter jeg har været en del af, var hjemmesiderne der var lavet dertil, set som at blive tilgået online. Men på grund af scoping og tid, lykkedes det aldrig at fuldende dette ønske. Derfor havde jeg fastlåst at denne gang skulle være anderledes.

I analyse delen af rapporten forklares mere i dybden om hvordan denne udfordring blev udført.

## 8. Løsningdesign

---

### 8.1 Flask

Programmet er opbygget omkring Flask.

Flask er et micro-web framework der giver udviklere flere værktøjer til at bygge en hjemmeside op med Python. Flask bruger WSGI og Jinja Templates til at yde diverse funktionaliteter som ville blive forklaret snarest.

---

### 8.2 HTML/CSS

Selvom alt backend sker gennem Python, skal det visuelle alligevel blive opstillet gennem HTML og CSS. Hjemmesiden bruger et sæt af objekter som gentager sig på hver side, og selve hovedindholdet bliver udskiftet ud fra brugerens interaktion med siden gennem Jinja.

Jinja bruges i dette projekt til to hovedfunktioner: Det første er til at “extend” html indhold fra et HTML-dokument til et andet. Det ville sige at det er muligt at stadig være på den samme side, men udskifte indhold som der bliver brug for det.

Den anden hovedfunktion er til at overføre variabler fra Python-delen af programmet og bruge dem direkte på HTML-siden uden at gå udenom ved at bruge Javascript. Det skal dog siges at Javascript også bruges til dele af siden, og det forklares nærmere i kodebeskrivelsen.

Til at behandle de adskillige objekter defineret på HTML-siderne, er der blevet skabt et “Style” CSS dokument, til at organisere og lave hurtige ændringer til elementerne på HTML-siderne. Fordelen ved at have et separat CSS dokument til elementerne, er at den bliver læst, hver gang en side bliver loaded. Derved gør dét det nemmere at lave små-ændringer uden at genstarte programmet.

---

### 8.3 WSGI

Under normale omstændigheder, er Python bygget til at udføre én opgave af gangen. Man kan bruge multithreading til at opdele udførelsen af koden til flere processor, men i Flask virker denne tilgang desværre ikke. Når en side bruges på nettet, skal den helst kunne gennemføre anmodninger fra mange brugere på én gang. Heldigvis bruger Flask WSGI til at ordne præcist dette.

WSGI, forkortelsen af Web Server Gateway Interface, er bygget præcist til at hjælpe hjemmesider bygget i Python til at behandle adskillige anmodninger på samme tid. Når en bruger sender en anmodning til serveren, bliver den anmodning sent til WSGI med instruktioner, og WSGI derefter eksekvere disse instruktioner samtidig i en container og sender dem tilbage til serveren i den korrekte rækkefølge. Koden der kræves udføres stadig på selve serveren, og derfor bliver der brugt processorkraft i at udføre disse

instruktioner, men flere instanser af instruktioner kan blive udført samtidig gennem WSGI.  
(WSGI.readthedocs.io) (Builtin.com)

---

## 8.4 API's

Hjemmesiden facilitere indsamlingen og opstilling af informationer, men selve kilden for informationen indsamles fra et tredje-parti.

Square Enix (SE), har lavet spillet FFXIV, men tillader ikke direkte adgang til data på deres servere, vedrørende deres markedsplads.

I stedet for, har en gruppe af dygtige programmører indsamlet disse informationer ved at opfange og formater de data-pakker der bliver sendt mellem SE's servere og spillernes klient, hvor en lang række data omkring hvert objekt kunne blive nedskrevet. Et eksempel på disse data kan ses på deres Github: <https://github.com/orgs/xivapi/repositories>

Projektet bruger to forskellige API'er. "XIVAPI" og "Universalis-API" og jeg ville her forklare deres omfang og hvordan de bruges i projektet.

XIVAPI er en Open-source API, samt en kollektion af spil-data og parsers der kan udtrække disse informationer fra klienten. (XIVAPI.com)

Selve API'en kan hente og søge efter spil-data, samt spiller-data, Free Company og Linkshell-information m.m.

Informationen bliver typisk hentet gennem en GET-Request og opstilles i JSON format. I forhold til projektets omfang, søges der efter information omkring et objekt via dens ID, som programmet har hentet via en lokal JSON fil.

Universalis er en åben API som man kan indhente oplysninger fra.

Markedsinformation bliver indsamlet og opdateret af brugere der åbner FFXIV's markedsplads i klienten, med et plugin/program kørende i baggrunden. Ud fra de indsamlede oplysninger, kan man bruge deres API til at indhente de relevante oplysninger og opstille dem som ønsket. F.eks. er det muligt at indsamle status på alle opslag af et objekt på markedet på den server der er relevant. Eller man kan hente salgs-historikken af et objekt, for at f.eks. opsætte en historisk-salgstrend-tabel. Alt data er opstillet i et JSON-format og et eksempel på informationerne dokumenteres i kodebeskrivelsen.

På grund af at disse API's er lavet ved at tage brug af en decompiler, er det usikkert hvorvidt brug af disse data kan ses som krænkelse af ophavsret og derved også sætte spørgsmål tegn ved, om brugen af API'en er i sammenhæng også er en krænkelse. (Plektonlabs, 2021)

---

## 8.5 Server-Hosting

Under udviklingen af programmet, kan man køre serveren lokalt og tildele den en IP. Dog uden nogen WSGI-containers, er den ikke bygget til at køre som et produktions-miljø. Dette er også tydeligt klargjort af programmet, hver gang det startes. Derfor, hvis dette produkt skal kunne skalere, er det nødvendigt at opstille et korrekt produktions-miljø. Det var muligt at opstille min egen WSGI-container og have en Raspberry som en altid-online host. Men eftersom at Cloud-services er blevet raffineret og billigere over årene, var det oplagt at tage brug af den service, så jeg skulle bruge mindre tid på opsætning og vedligeholdelse, så jeg kunne fokuserer på andre ting.

Den første service jeg undersøgte var Amazon Cloud Series (AWS). Deres bibliotek af cloud-services er velkendt for deres bredt udvalg af services, men efter at have dykket ned i det med at opsætte en Ubuntu server på deres platform, var der flere problemstillinger der opstod. Da jeg stadig ikke havde en WSGI-container eller en måde at sørge for sikkerhed på hjemmesiden via certifikater og sikre forbindelser gennem HTTPS.

Derved kom jeg til en ny mulighed; Pythonanywhere.

Pythonanywhere er bygget præcist til at køre Python applikationer i cloud og havde en større mængde af features til Flask der hjalp med nem opsætning og vedligeholdelse. Ved at bruge deres egen domæne, kunne man få certifikater og HTTPS indbygget uden ekstra omkostninger eller opsætning, men når produktet skifter fra udvikling til produktions-miljø, skal der overvejes at tage brug af egen domæne.

Med en “Web Dev”-konto for \$12 om måneden, havde jeg mere end rigeligt af CPU-tid til at bruge til projektet.

---

## 8.6 SSH

Overførslen af dev-miljøet til Pythonanywhere kunne ikke gøres med en simpel direkte fil-overførsel på grund af at de ikoner der bruges til at vise en visuel repræsentation af objekterne, tager omkring 2.1GB af plads og er derfor mere end hvad Pythonanywhere tillader. Derfor måtte jeg overføre dem gennem SSH. Mere specifikt, gennem Github SSH-metoden.

Git funktioner er indbygget i PythonAnywhere bash konsollen og med det kan man nemt hente og uploadere Git-depotet med et par kommando-linjer. Da mit projekt er sat som privat på Github, kræver det mere sikre metoder for at få adgang. Efter 2021 kan man ikke længere bruge brugernavn og adgangskode til at få adgang, men i stedet kan man opsætte en SSH-Key.

The screenshot shows the GitHub 'SSH keys' settings page for the user 'Erunoma'. The left sidebar has a 'SSH and GPG keys' section selected. The main area lists three SSH keys:

- pythonanywhere1**: SHA256:QPkCoYwgh598dc/VSR4Tyc8cd19UkB88AxB/F1kRY0 (SSH). Added on Nov 3, 2023. Last used within the last 7 months — Read/write. Delete button.
- sapphire\_ssh**: SHA256:1laUAMCva++5yzjU1vmURNyB1YXOIttcKmpgdmTdQA (SSH). Added on May 9, 2024. Never used — Read/write. Delete button.
- sapphire\_server\_pythonanywhere**: SHA256:oNxPYbf7CAsDoD+R5ZD8vZ2EcNf3qSIXouepwd6IovE (SSH). Added on May 9, 2024. Last used within the last week — Read/write. Delete button.

At the bottom, there is a note: "Check out our guide to [connecting to GitHub using SSH keys](#) or troubleshoot [common SSH problems](#)."

[Figur 9- Github Settings]

På PythonAnywhere konsollen, blev der brugt kommandoen ssh-keygen med en identifier, til at lave en unik key. Efter key'en var lavet, kunne man se indholdet ved at bruge "cat" kommandoen. Denne lange string indsættes i feltet under SSH Keys i Github's bruger-indstillinger.

Efter adgangen er etableret, kan man derefter bruge "git clone" til at hente hele programmet.

## 8.1 Kodebeskrivelse

I dette afsnit forklares den gennelige opsætning af koden og derefter hvordan hver hoved-funktion hænger sammen med det overordnet program.

Programmet starter i main.py, hvor Flask bliver initialiseret og hjemmesiden bliver opsat. I et lokalt miljø, startes hjemmesiden med ‘Flask.run’, som er defineret her som ‘app’.

Under udvikling, ønskes der ikke at andre enheder kan tilgå siden, så host-IP’en er sat til localhost, og port 43435. Localhost holder serveren indenfor enheden, mens porten kan i principippet være hvilket som helst åben port på netværket.

```
16     app = Flask(__name__, template_folder='html')
17
18     def start():
19         try:
20             if is_local==True:
21                 app.run(host='localhost', port=43435)
22             else:
23                 app.run(host='0.0.0.0', port=43435)
24
25         except:
26             print(print_exc())
27             exit()
```

[Figur 10 - main.py]

Programmet har adskillige funktioner der kan tilgås med en GET-Request ved at indtaste den korrekte URL eller bruge navigations-baren på venstre side af hjemmesiden. Hver hovedfunktion har dens egen path, men det er muligt at tilgå individuelle objekter, ved at indtaste den rigtige search-query i url’en. Så links kan gemmes for at hurtigere kunne slå ting op.

Den første hovedfunktion er hvor man kan søge efter objekter i spillet og få information omkring hvilket opslag der er på den individuelle server man vælger. Denne hovedfunktion kaldes for ‘Item Search’



[Figur 11 - Hjemmeside Hovedside]

### 8.1.1 Item Search

Når der søges efter et objekt via søge-feltet, får brugeren først en liste af alle objekter der kan søges efter ved at kigge igennem 'item\_list' variablen, som er en liste af alle engelske navne for objekterne i 'item\_id\_information.json' filen. Derefter sorteres de i real-time ud fra hvad brugeren skriver. Hvert objekt i listen er automatisk generet som et 'li' objekt.

Når brugeren kan se hvilket objekt de søger efter, kan de klikke på den specifikke objekt og blive nавигeret til en ny side, hvor URL'et er generet ud fra objektets navn og hvilket server man søger på.

Imens denne side bliver loaded, sender programmet en API Request til Universalis og efterspørger opslag-information ud fra det tidligere defineret query.

Når opslagende er fundet, søger programmet også for ID'et af objektet via funktionen 'get\_item\_id\_from\_name()', som søger igennem item\_id\_information.json og finder ID'et ud fra navnet. Når ID'et er fundet, startes en ny API Request til XIVAPI, for ikon-ID'et. XIVAPI sender tilbage en json hvor det ligger i. Til sidst bliver der via 'find\_picture\_path()' beregnet ud fra ID'et, hvor ikonet ligger i mapperne som er gemt lokalt. Ikonerne er kategoriseret efter ID-omfanget, og er samlet sådan da det først blev downloadet fra XIVAPI.

Derefter opsættes alt information og sendes til siden i render\_template. Som derefter kan opsættes i HTML'en med Jinja.

```

87 def find_picture_path(icon_id):
88     #Pathing funktionen her er blevet oversat delvist til Python via ChatGPT fra kilde: https://xivapi.com/docs/Icons
89     icon_id=str(icon_id)
90     print(f"Searching for path to item with Icon ID: {icon_id}")
91
92     # Check the length of icon_id
93     if len(icon_id) >= 6:
94         icon_id = pad(5, "0", icon_id)
95     else:
96         icon_id = '0' + pad(5, "0", icon_id)
97
98     # Now we can build the folder from the padded icon_id
99     if len(icon_id) >= 6:
100         folder_id = icon_id[0] + icon_id[1] + icon_id[2] + '000'
101     else:
102         folder_id = '0' + icon_id[1] + icon_id[2] + '000'
103
104     # Concatenate the folder_id and icon_id to form the path
105     path = 'static/icons/' + folder_id + '/' + icon_id + '.png'
106     print(path)
107     return path

```

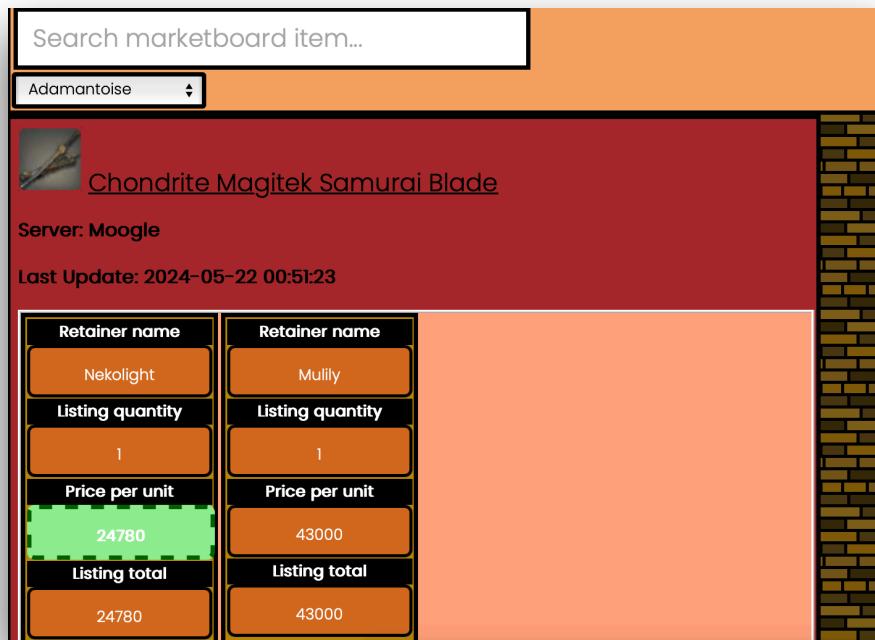
[Figur 12 - item\_search.py]

```

36 def item_lookup(item_id, server):
37     listings=[]
38     try:
39         json_item=requests.get(f"https://universalis.app/api/v2/{server}/{item_id}")
40
41         if json_item.status_code==200:
42             print("got item request")
43             item_json=json_item.json()
44             last_update_time=last_update(item_json)
45             #pprint(item_json)
46
47             for listing in item_json["listings"]:
48                 listing_data={"retainer_name":listing["retainerName"],"listing_quantity":listing["quantity"],
49                               "price_per_unit":listing["pricePerUnit"],"listing_total":listing["total"]}
50
51                 listings.append(listing_data)
52             #pprint(listings)
53
54
55     except:
56         print("Failed to get result")
57         print(print_exc())
58
59
60     return [listings, last_update_time]

```

[Figur 13 - item\_search.py]



[Figur 14 - Item Search resultat]

### 8.1.2 Crafting calculator

Denne hovedfunktion fungere næsten ligesom Item Search, men når et objekt søges efter, bliver der først lavet en GET-Request til XIVAPI for objektets 'RecipeID', som derefter bliver lavet et nyt Request til. I den Recipe ID JSON, finder man information omkring antal af hver komponent og hvilket ID de har. Der findes en god del andet information i disse JSON's, men til projektet er det de informationer der bruges. I den JSON kan de findes under [GameContentLinks][Recipe] og [ItemResult].

Når alle komponenter er fundet, laves der et Item Search for hver af disse, så deres opslags-information kan bruges til at finde den mindste pris.

Deres ikoner bliver også fundet ud fra funktionen tidligere beskrevet.

Navne, mindste pris, ID og Ikon er blevet samlet i hver deres liste, men er Zippet sammen til sidst i én liste. Kan ses på linje 74 på figur 16

```
"GameContentLinks": {
    "GCSupplyDuty": {
        "Item07": [
            88
        ]
    },
    "GilShopItem": {
        "Item": [
            "263056.1"
        ]
    },
    "Recipe": {
        "ItemResult": [
            5029
        ]
    }
},
```

[Figur 15 - XIVAPI /item/ - Resultat]

Når alle komponenternes data er blevet indsamlet, laves der en hurtig beregning for at finde total prisen til at lave objektet der først var blevet søgt efter. Derefter bliver det opstillet i HTML med Jinja. Hvert komponent og det originale objekt får tildelt et generet link til deres Item Search side, hvis brugeren ønsker at få mere information omkring de individuelle dele.

```

40     component_ids=[]
41     ingredient_amounts=[]
42     #Adds the amount into a list
43     for key in ingredient_values:
44         ingredient_amounts.append(ingredient_values[key])
45
46     #For each ingredient, searches for the ID in the recipe and adds the ITEM ID to the list
47     for key in ingredient_values:
48         ingredient_search_number=int(key.split("AmountIngredient")[1])
49         print(f"Search number are as follows: {ingredient_search_number}")
50         component_ids.append(recipe_json[f"ItemIngredient{ingredient_search_number}"]["ID"])
51     component_names=[]
52
53     #Finds and adds every ingredient name to a list
54     for component in component_ids:
55         component_names.append(find_component_name(component))
56     print(f"Component ids: {component_ids}, ingredient amounts: {ingredient_amounts}")
57
58     #Finds and adds every lowest price of the item to a list
59     components_lowest_price=[]
60     for component in component_ids:
61
62         item_listing=item_search.item_lookup(component,server)
63
64         lowest_price=item_search.find_lowest_price_per_unit(item_listing[0])
65         components_lowest_price.append(lowest_price)
66
67     #Finds and adds every Icon path to a list
68     components_icon_paths=[]
69     for component in component_ids:
70
71         icon_id=item_search.find_item_icon_id(component)
72         print(f"Icon ID: ")
73         components_icon_paths.append(find_picture_path(icon_id))
74         component_information_tuple=[x for x in zip(component_ids,ingredient_amounts,component_names,
75                                         components_lowest_price, components_icon_paths)]
76
77     print(f"This is the final info: {component_information_tuple}")
78     #Returns all of this information in a big list
79     return component_information_tuple

```

[Figur 16 - crafting\_calculator.py]

### 8.1.3 Character Search

Den sidste hoved-funktion er en kollektions-oversigt med marked data.

Siden man starter på har et søgefelt, hvor man kan søge efter en spiller-karakter. Et karakter-navn er to ord, mellem 2-15 bogstaver i hver med 20 bogstaver tilladt i alt, med kun få symboler tilladt.

Søge-feltet er beskyttet med Regex ud fra disse konfigurationer: “^ [a-zA-Z ]+ \$”

Når der søges via en POST, søges der efter karakterer med navne der matcher. Via scraping metoden importeret med BeautifulSoup modulet, samles alt HTML data fra FFXIV’s officielle hjemmeside, hvor de har en søgemaskine. Dette gøres ved at indsætte navnet ind i url’et: ‘<https://na.finalfantasyxiv.com/lodestone/character/?q={name}&worldname={server}>’.

Når HTML dataen er indsamlet til bs4 objekt, søger programmet HTML’en efter alle <div> objekter med ‘class’ navnet: ‘entry’.

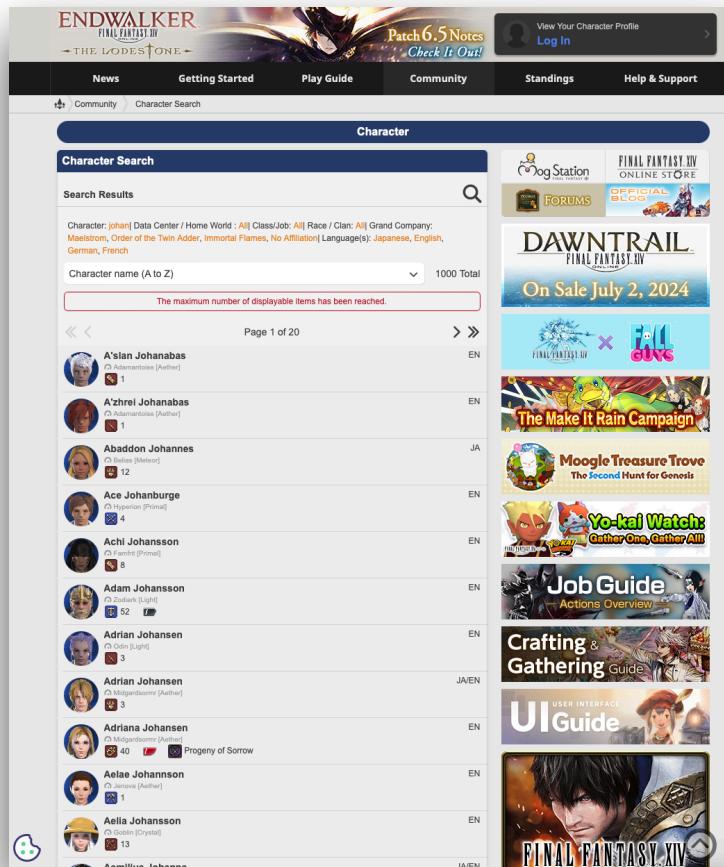
```

11 def search_lodestone_character_list(name, server):
12     char_info=[]
13     #Linje 14 er skrevet delvis af ChatGPT
14     if bool(re.search(r"\s", name))==True:
15         print("This name has a space")
16         name=name.replace(" ","+")
17     cleaner=re.compile('<.*?>')
18     print(f"url: https://na.finalfantasyxiv.com/lodestone/character/?q={name}&worldname={server}")
19
20     if server:
21         page=requests.get(f"https://na.finalfantasyxiv.com/lodestone/character/?q={name}&worldname={server}")
22         if page.status_code!=200:
23             raise Exception
24     else:
25         page=requests.get(f"https://na.finalfantasyxiv.com/lodestone/character/?q={name}")
26         if page.status_code!=200:
27             raise Exception
28
29
30     #Linje 31-32 er taget fra mit tidligere-lavet Scraping program fra praktikken
31     soup = BeautifulSoup(page.content, "html.parser")
32     clean_text=re.sub(cleaner, '', str(soup))
33
34     #print(clean_text)
35     #print(soup)
36     html_listings=soup.find_all("div", {"class": "entry"})

```

[Figur 17 -lodestone\_scrap.py]

Her ses hvordan det originalt ser ud på siden (Søgeord: Johan):



[Figur 18 - Lodestone Character Search]

Derefter udtrækkes tre variabler som bruges i programmet. Navnet, karakterens ID, og href til karakterens ikon.

Disse informationer bliver vist på siden som resultater således:



[Figur 19 - Hjemmeside Character Search]

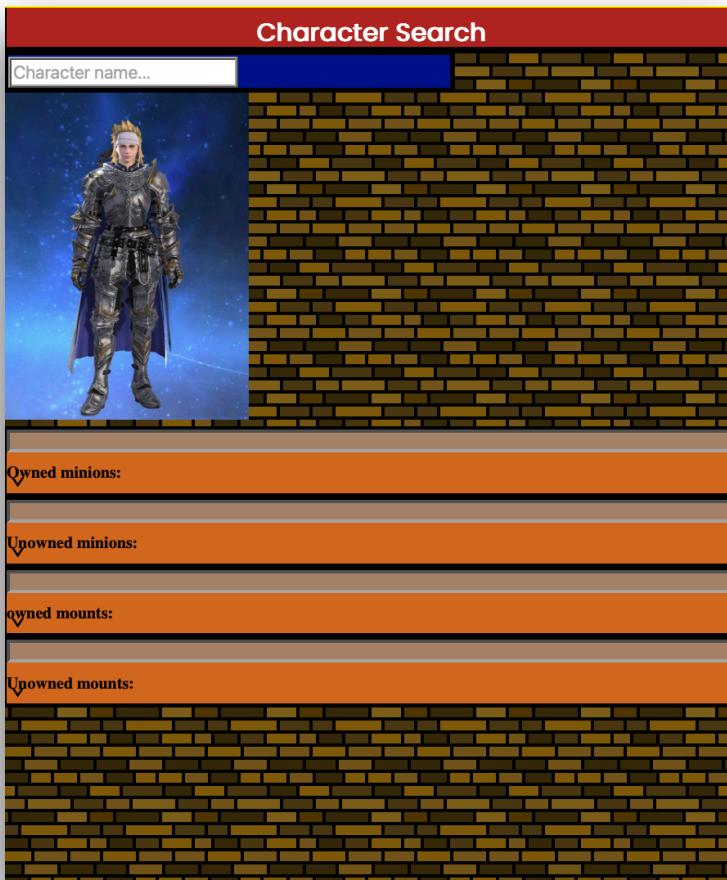
Når brugeren klikker på en bestem karakter, tages de videre til kollektions-siden.

Kollektions siden er her hvor brugeren kan se karakterer kollektion af Mounts og Minions<sup>3</sup>.

Når en karakter indlæses, sker der to konkrete funktioner.

---

<sup>3</sup> Mounts er noget spilleren kan ride på, og Minions er som små dyr der følger med spilleren rundt



Den første er at den allerede indsamlet information omkring mounts og minions fra CSV filerne i /data mappen, bliver sat i lister til at sammenligne med. CSV filen har fire variabler per mount/minion: Navn, ID, Icon path, og en boolean til at vise om den kan sælges eller ej på markedet med et True eller False.

Det andet der sker er at karakterens Lodestone side bliver scrapet ud fra karakterens ID. Herefter findes der to data-punkter der bruges til siden: Karakterens fulde profilbillede og hvilket server de kommer fra. Billedet bliver vist først når brugeren kommer ind på siden, og serveren bliver brugt til at lave et link til Item Search, der også kræver en server-variabel i dens query.

[Figur 20 - Hjemmeside Karakter Profil]

Derefter scrapes der igen til at søge på to hjemmesider: Karakterens minions og karakterens mounts. Hver har deres egen side og derfor må gøres individuelt, men størstedelen af HTML koden på siden er det samme, så fremgangsmåden ændre sig ikke meget.

Med HTML-dataen, søges der efter navnet på minion/mount via dens href.

På grund af den måde som Square Enix har opsat denne hjemmeside på, er der ingen navne på dem der kan nemt tages brug af. I stedet, så fungere siden sådan at når man holder musen over et ikon, laves der javascript i baggrunden der downloader og viser en boks med informationen omkring denne mount/minion. Desværre indlæser bs4 kun den html data der findes når siden indlæses og kan ikke simulere kliks.

Men hvert ikon har en href i sig, der kan navigeres til via GET-requests. Det er derfra der trækkes navnet fra. Dog har programmet allerede 'href' datapunktet i CSV'en, så derfor sammenlignes disse data i stedet for, og derved undgår man at sende GET-requests.

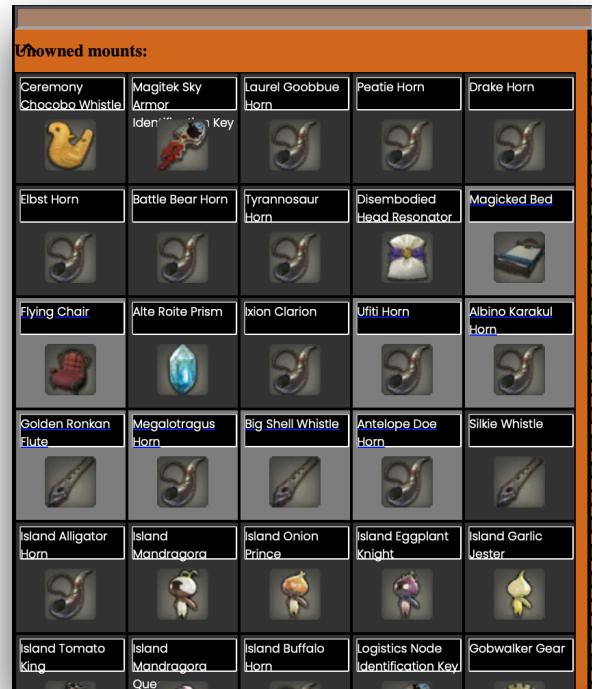
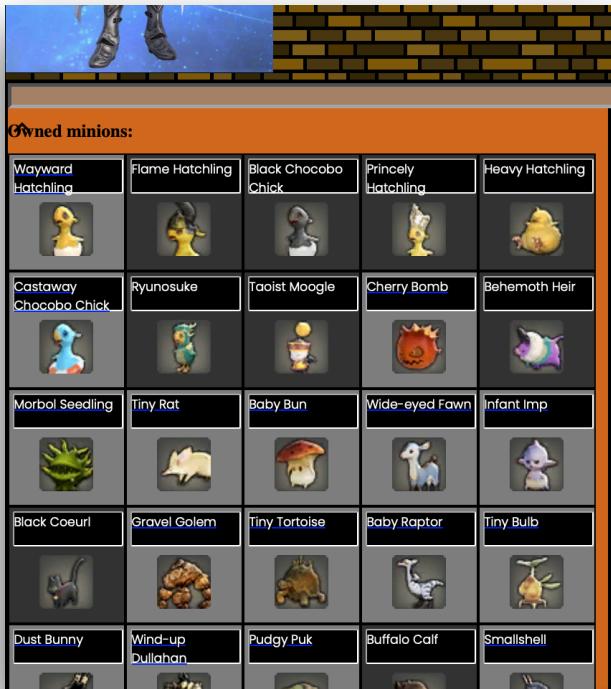
Når navnene er fundet, sorteres programmet alle minions og mounts i lister ud fra om de kan sælges eller ej.

Der bliver også tilføjet deres ikon ud fra tidligere-diskuteret funktion.

Derefter sorteres de ud fra om karakteren har minion/mount i deres kollektion, eller om de ikke har.

Når alle disse ting er gjort, indlæses siden for brugeren.

Her kan brugeren åbne for hvert vindue, for at tjekke deres kollektion. Dem der ikke kan sælges eller købes, er markeret med mørke-gråt, mens dem som kan er markeret med en lysere grå og kan klikkes på der linkes til Item Search siden for det objekt.



[Figur 21 & 22 - Hjemmeside Karakter Profil Samling]

## 8.1.4 Diverse kode-håndtering

### Pprint

Pprint-modulet bruges i koden under udviklingen til at få et bedre printout til konsollen, som er nemmere at læse. Når man normalt printer til konsollen, er det typisk i en lige linje uden mellemrum eller paragraf-opdeling. Med Pprint, opdeles f.eks. JSONs ud fra deres dictionary opdeling.

```
from pprint import pprint
```

[Figur 23 - pprint import]

### Traceback

Traceback-modulet bruges i koden til at få et bedre printout af fejl der sker under debugging. Det er ikke kun for at få mere information, men også for at definere hvilket fejl-beskeden bliver sendt til konsollen, som derved kan forbedre min egen evne til at pin-pointe præcist hvad fejlen betyder.

```
from traceback import print_exc
```

```
2024-05-23 07:26:55,613: Traceback (most recent call last):
2024-05-23 07:26:55,613:   File "/home/Erunoma/sapphire_avenue_toolkit/Main/lodestone_scraper.py", line 156, in get_mount_collection
2024-05-23 07:26:55,613:     mount_item_name=str(mount_item_a.get("data-tooltip"))
2024-05-23 07:26:55,614: AttributeError: 'NoneType' object has no attribute 'get'
```

[Figur 24 & 25 - Traceback Eksempel]

## 8.2 Visuel Design

### 8.2.1 Inspiration

En brugers interaktion med siden er lige så vigtig som det der foregår i baggrunden, derfor er det vigtigt også at fokusere på det visuelle og brugerfladen generelt.

Final Fantasy er en spil-serie der går helt tilbage til 1987, og dens visuelle stil dengang bruger det vi i dag kalder for “pixel-art”, da opløsningen af hver tekstur og karakter var tegnet med få pixels. Typisk omkring 16x16 eller 32x32. (Wikipedia.com)

Personligt synes jeg at pixel-art har en vis charme. Min egen grafiske evner er ikke på niveau med dem som er uddannet i design eller lign., men pixel-art er nemmere at arbejde med og det er hvad jeg kender bedst til fra mine tidligere projekter. Så besluttede mig tidligt at bruge denne stil.

Ud fra Final Fantasy XIV: Encyclopedia of Eorzea Vol.1, er der beskrevet et stræde i gaderne af den fiktive by Ul'dah der ligger midt i en ørken, hvor folk kommer for at sælge og købe adskillige produkter fra nær og fjern. Boder er opstillet på række på begge side af gå-gaden ned langs det lange strøg en form af en halvmåne. Mønstre er indskrevet i firkantet fliser der markerer stien, mens der mellem gadelamperne hænger silke-bannere i adskillige farver. Strøgets navn er The Sapphire Avenue Exchange. (Oda B., 2022)

Denne beskrivelse inspirerer stilen til hjemmesiden, farve-valget og navnet.

Til at lave det visuelle, gjorde jeg brug af programmet Piskel, et browser-værktøj der lader én lave pixel-art med nemhed.

Alt visuelt kan ses i bilag og i programmets source-kode.

## 9. Test af løsning

### 9.1 Interne Tests

Under udviklingen, efter hver sprint, testede jeg hjemmesidens stabilitet ved at have flere forskellige enheder og browser-faner forbundet til siden på samme tid. Det havde til formål at simulere et normalt produktionsmiljø, hvor flere brugere var forbundet. På den lokale test gik det hurtigt op for mig at serveren, i dens daværende tilstand, ikke kunne foretage sig flere requests fra flere brugere på én gang og gav én af dem en error.

Ud fra dette indså jeg at serveren skal have funktionen i at køre flere processorer på én gang, så den ikke venter for lang tid.

Men løsningen fandt sig selv, da serveren blev opsat på PythonAnywhere, som havde en indbygget WSGI-funktion, der automatisk opsatte den ønskede funktion uden ekstra input fra mig.

I fremtiden ville det være en god idé at opsætte sådan en funktion selv.

Derudover, har jeg inviteret adskillige bekendte til at prøve siden og give feedback.

## 9.2 Udførelse af Accepttest

ID	Krav	Prioritet	Accepttest	Status
1	Brugeren kan via hjemmesiden søge efter et samleobjekt og får et nummer tilbage af den billigste pris der findes på markedet.	1	<b>Resultat:</b> Via Item Search søger man efter et objekt, og kort tid efter vises alle opslag (op til 100) der lige nu er registreret hos Universalis. Hvert opslag har navnet på sælgeren, hvor mange der sælges og prisen for hver samt total.	Passed
2	Brugeren kan via hjemmesiden søge efter et objekt der kan sælges, og få oplysninger omkring materialerne der bruges til at lave det, og den estimeret pris, hvis materialerne skulle købes via markedspladsen.	1	<b>Resultat:</b> Via Crafting Calculator søger man efter et objekt og efter et længere stykke tid, vises objektet, med ikon, server og dens billigste pris på markedet og hvornår det senest var opdateret. Nedenunder ses opskriften med alle objekter der kræves for at lave den, med mængden og deres billigste pris på markedet. Til sidst er der sammenlagt den totale sum af bekostningen for at lave det objekt.	Passed
3	Hjemmesiden skal være beskyttet mod misbrug af API'en	1	<b>Resultat:</b> Der er i øjeblikket ingen restriktioner på API forbrug på serveren selv. Der er en grænse for API'en fra en enkelt IP, så hvis serveren får for mange API requests sendt på kort tid, kan serveren blive forhindret i at sende flere requests i et stykke tid. Dette kan fikses ved at sætte en grænse requests ud fra brugerens IP	Failed
4	Hjemmesiden skal kunne tilgås fra internettet	1	<b>Resultat:</b> Siden er sat op via Pythonanywhere, som hoster fra en Amazon Web Services (AWS) linux maskine på IP: 35.173.69.207 (Da dette skrives), og kan tilgås fra flere netværks via <a href="http://erunoma.pythonanywhere.com">erunoma.pythonanywhere.com</a>	Passed
5	Hjemmesiden skal kunne bruges både i mobil-format og desktop-format	2	<b>Resultat:</b> Afhængig af vinduets størrelse, justere hjemmesiden sig automatisk til skærmen ved at bruge %-Width/Height i CSS.	Passed
6	Serveren er hostet på en Raspberry PI eller via en hosting side	2	<b>Resultat:</b> Serveren er hostet hos PythonAnywhere, hvor en månedlig betaling giver adgang til 4.000 CPU sekunder og 5.0GB af plads. PythonAnywhere går igennem Amazon Web Services (AWS) og er derfor online 99.9% af tiden.	Passed

ID	Krav	Prioritet	Acceptest	Status
7	Brugeren kan slå op deres karakter og få en oversigt over deres samling. I deres samling kan de klikke på et objekt og få informationer omkring markedet om dette objekt.	2	<b>Resultat:</b> Via Character Search kan brugeren søge efter et navn der passer med FFXIV's karakter-navn-regex. Brugeren kan vælge ud fra de karakterer der matcher navnet og derefter kan deres profil ses. Her er både et fuldt billede af karakteren og deres minion og mount samling. Hver er opdelt i deres egen bokse, og de samleobjekter der kan sælges kan der trykkes på, med et link til deres Item Search side.	Passed
8	Hjemmesiden skal være responsivt og virke professionelt	3	<b>Resultat:</b> Der er grafiske elementer der har til formål at referer til et specifikt område i spillet der har at gøre med markedet. Når programmet henter informationer, bliver der vist et "loading"-ikon indtil den nye side er klar. Ventetiden kan dog være flere sekunder afhængig af situationen. Men det tager aldrig længere end 20 sekunder.	Passed

Showcase Video:



## 10. Implementering af løsning i drift

Dette program ville kunne implementeres enten som et stand-alone program som vedligeholdes af firma eller som en del af en større ‘pakke’ eller ‘hjemmeside’. Hvis dette produkt skulle bruges i sammenhæng med et nyt startup, er der flere ting der skal forbedres og implementeres, før der kan være chance for at tiltrække brugere og i sammenhæng, profit, såsom visse dele af det visuelle og klargøring hvilke oplysninger der bliver gemt på serveren. Der er dog i øjeblikket ingen data der bliver aktivt gemt, så dette punkt ville ikke være vigtigt på nuværende situation.

Der er under dette projekt ikke fokuseret på det juristiske, men projektet er lavet med respekt for brugerene, API-udviklerne og Square Enix. Dog kan der være gode grunde til at undersøge dette punkt nærmere i fremtiden.

Indkomst kan skabes enten via donationer fra tilfredstillede brugere, eller ved at placere reklamer på siden. Programmet skal blive langt større, før det føles som en god idé at have en form for medlemskab for visse funktioner.

## 11. Praktisk projektplanlægning og ledelse

### 11.1 Gantt/Kalender

Til projektets fremgang blev der brugt en række værktøjer til at sørge for bedre tids- og arbejds-estimering, samt at skabe overblik over projektets pipeline.

Til at danne et overblik over arbejdsdage og sprints, opstille jeg en personlig kalender i Numbers, lavet ud fra min egen template der indeholder alle arbejdsuger som projektets deadline står indenfor.

Kalenderen, som ses på figur 26, har to hovedfunktioner; den første er at definere arbejdstiden fordelt over ugens dage. Hovedreglen var at arbejde 25 timer om ugen på projektet, fordelt på de syv dage. Typisk blev der sat fem timer af per hverdag, men til tider kunne der opstå komplikationer i programmet, og derved måtte nogle timer blive rykket rundt.

Den anden hovedfunktion var til at definere projektets sprints. Disse var fordelt ud fra hvad jeg personligt kunne minimum nå at implementere i produktet, mens samtidig færdiggøre rapporten inden for rimelig tid. Derfor blev sprints sat op med en 3-dags og så til 2-dags sprint. Fra mandag - onsdag, og torsdag-fredag\*. På den måde, havde jeg mulighed for at fordybe mig på områder der kræver ekstra tid, og så raffinere det i dagene op til weekenden.

**Uge 19**

	April 29	April 30	May 1	May 2	May 3	May 4	May 5
	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
<b>Oliver</b>	12 - 17	12 - 17	12 - 17	12 - 17	12 - 17	X	X
<b>Sprints</b>	1	1	1	2	2		

**Uge 20**

	May 6	May 7	May 8	May 9	May 10	May 11	May 12
	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
<b>Oliver</b>	15:30 - 20:30	12 - 17	X	18-22	12 - 18	3 Hours	2 Hours
<b>Sprints</b>	3	3	3	4	4		

**Uge 21**

	May 13	May 14	May 15	May 16	May 17	May 18	May 19
	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
<b>Oliver</b>	10 - 15	12 - 17	12 - 17	10 - 12, 16 - 19	12 - 17	X	X
<b>Sprints</b>	5	5	5	6	6		

**Uge 22**

	May 20	May 21	May 22	May 23	May 24	May 25	May 26
	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
<b>Oliver</b>	12 - 17	12 - 17	12 - 17	12 - 17	12 - 17	X	X
<b>Sprints</b>	7	7	7	8	8		

**Uge 23**

Hand-in Date

	May 27	May 28	May 29	May 30	May 31	June 1	June 2
	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
<b>Oliver</b>	12 - 17	12 - 17	12 - 17	12 - 17	12 - 17	12 - 17	12 - 17
<b>Sprints</b>	9	9	9	10	10	11	11

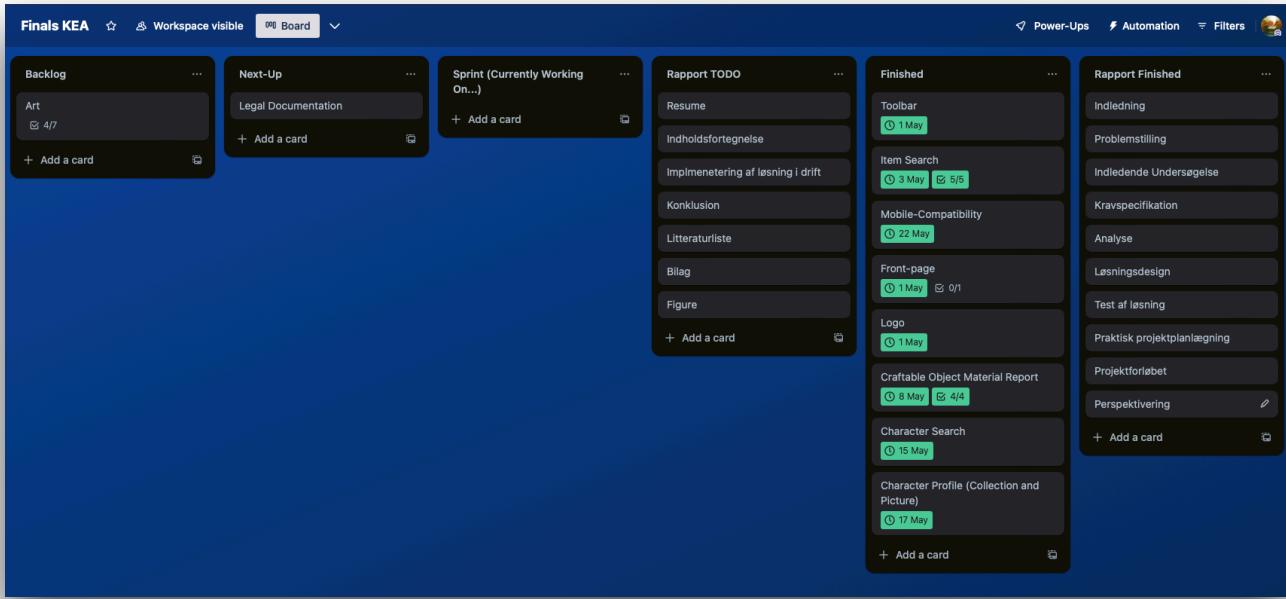
Programming	Final Crunch	Writing	Research/ Writing	Free

[Figur 26 - Gantt/Kalender]

## 11.2. Scrum/Trello

Til at holde styr på alle features og dele af projektet, opsatte jeg et Scrum-board i Trello. Her kunne jeg indsætte alle nødvendige features og kategorisere/priorititere dem som nødvendigt. Med mulighed for at sætte deadlines på hvert kort, kunne jeg bedst skabe miljøet til bedst produktivitet og fokus.

[Figur 27 - Scrum,/Trello]



### 11.3. Profit og fremgang

Legaliteten omkring projektet er uden for scopet af denne rapport, men som en uafhængig hjemmeside, kan der være mulighed for at indsamle kapital til enten at profitere, eller i det mindste, betale for omkostningerne ved at drive en hjemmesiden med minimal support.

En tredje mulighed er at sælge hjemmesiden, inklusiv source-kode til en anden.

## 12. Konklusion

Ud fra de opstillede krav og med resultatet af de test der er blevet udført, kan det konstateres at produktet kan udføre unikke markeds-funktioner der kan tilføje til brugerens profitering i spillet Final Fantasy XIV.

Derudover er det bevist at der findes et marked for disse typer af værktøjer, som der kan bygges en forretningsmodel op omkring. Med brugerundersøgelsen blev det klart at værktøjets funktioner skal dække interessen af flere typer af spillere, og hvis det bliver muligt at samle et større datasæt i fremtiden, ville der kunne gøres en mere dybdegående analyse end hvad der var muligt i denne rapport.

Med et større budget kunne denne prototype udvikle sig til at have uforglemmelig indflydelse på resten af markedet, både i FFXIV og uden for det. Der er dog steder hvor produktet kan forbedres, nævnligt i den visuelle opsætning og hastigheden af backend funktioner. Desuden er der vigtige juridiske problemstillinger som skal tackles, men erude af scope for denne tekniske rapport.

Med få justeringer og videreudvikling, kan produktet blive brugt aktivt af tusindvis af spillere i fremtiden.

## 13. Projektforløb

Dette var mit første store projekt, som jeg har valgt at gøre alene. Der var flere grunde til det, men generelt var det mest en følelse af at hvis jeg ønsker at specialisere mig i et felt, så skulle det være programmering, og det jeg ville arbejde indenfor det, kan være svært at bygge rundt om en fælles idé, hvor potentielle andre fag kunne indgå.

Jeg føler at jeg har arbejdet godt med projektet, og har nået at lave et tilfredstillende produkt. Der er selvfølgelig tusindvis af ændringer og forbedringer jeg havde ønsket at lave, men rapporten kom først. Det har været svært at vægte tidsopdelingen mellem program og rapport, men ved at fokusere mere på rapporten, har jeg kunne scope programmet til hvad jeg har været komfortabelt med.

Til trods for det, har projektet kørt på skinner uden de værste bump på vejen. De mål jeg havde sat for projektet blev udført indenfor de sprints de var allokeret, med tid til overs til at starte på nogle sprints tidligt.

Når man arbejder alene, er der mange ting man skal være opmærksom på. Der er ingen der kan dele byrden, man er selv om det. Til tider er det frigivende, men andre gange kan det være stressende at huske alle dele. Fortrækker klart at arbejde på større projekter med kammerater, men der har været nogle unikke vækstmuligheder med dette projekt.

## 14. Perspektivering

I løbet af projektets fremgang, deltog jeg i samtaler på diverse forums omkring Universalis og XIVAPI. Der var andre kloge programmøre der havde lavet deres egen værktøjer der hængte sammen med markedet eller spillet. Hver havde deres egen vision, og det gik op for mig, desværre for sent, hvordan jeg kunne have struktureret mit program til at undgå ventetiden på requests og pointers til generel optimering af programmet. @indopan på Discord fortalte mig omkring hans måde at håndtere store mængder af data via en API. Ved at køre et separat program, som ofte lavede requests til API'en, kunne han have en opdateret database med alt informationen der kræves til hans side. Derfor skulle hjemmesiden selv ikke lave nogle requests, og alt blev hentet og indlæst hurtigt lokalt. Størrelsen på alt det data, blev jeg fortalt, var på 12GB.

Det er klart en bedre måde jeg kunne gøre siden hurtigere på.

Den visuelle design del af programmet er ikke hvad jeg kalder for kunst, men det er hvad jeg havde færdighederne til at gøre. Måske skulle jeg i stedet have brugt kun CSS til at designe siden... Jeg brugte en god CSS til dette projekt, og jeg føler at jeg har taget et større skridt mod at være mere komfortabelt med det.

Derudover ønskede jeg også at kunne lave mere dybdegående undersøgelser om det virtuelle marked, da det er et fascinerende emne. Men det er været en byrde at holde anslag indenfor de krævede mængder, så måtte gøre det ekstremt kort.

## 15. Litteraturliste

*APIs and ethics: What are the ambiguities?* (2021) *PlektonLabs*. Kan læses: <https://www.plektonlabs.com/api-and-ethics-what-are-the-ambiguities/> (Tilgået: 24 April 2024).

*Final fantasy (video game)* (2024) *Wikipedia*. Kan læses: [https://en.wikipedia.org/wiki/Final\\_Fantasy\\_\(video\\_game\)](https://en.wikipedia.org/wiki/Final_Fantasy_(video_game)) (Tilgået: 19 Maj 2024).

*Grafana Universalis*. Kan læses: <https://monitor.universalis.app/d/3PpqjXv4k/universalis?orgId=1&refresh=30s> (Tilgået: 20 Maj 2024)

*GE tracker: Old school runescape's most advanced flipping and money making tool, OSRS Flipping - Grand Exchange Money Making - GE Tracker*. Kan læses: <https://www.ge-tracker.com/> (Tilgået: 16 Maj 2024).

McArthur, V. et al. (2012) ‘Knowing, Not Doing: Modalities of Gameplay Expertise in World of Warcraft Addons’, *Extended Abstracts on Human Factors in Computing Systems*, side. 101–110.

*MMO definition & meaning*, *Dictionary.com*. Kan læses: <https://www.dictionary.com/browse/mmo#> (Tilgået: 24 Maj 2024).

Oda, B. et al. (2022) *Encyclopaedia Eorzea Vol 1.*, CA: Square Enix Books, an imprint of the Book Publishing Division of Square Enix, Inc. side. 128 - 139

Rapp, A. (2018) ‘Social game elements in World of warcraft: Interpersonal relations, groups, and organizations for Gamification Design’, *International Journal of Human–Computer Interaction*, 34(8), side. 759–773.

*Regarding third-party tools* (2022) *FINAL FANTASY XIV, The Lodestone*. Kan læses: <https://eu.finalfantasyxiv.com/lodestone/topics/detail/68f4c4d3bdea34f732ee8db3425122b8b8d6ec3b> (Tilgået: 08 Maj 2024).

*R/FFXIV on reddit: What exactly are ‘Third party tools’?* Kan læses: [https://www.reddit.com/r/ffxiv/comments/umouq5/what\\_exactly\\_are\\_third\\_party\\_tools/](https://www.reddit.com/r/ffxiv/comments/umouq5/what_exactly_are_third_party_tools/) (Tilgået: 08 Maj 2024).

*PAX East - Journey to Dawntrail: The 10 Year Adventure of FINAL FANTASY XIV* (2024) *Twitch*. Kan ses: <https://www.twitch.tv/videos/2099153074> tid: 16:31 (Tilgået: 12 May 2024).

*Universalis*. Kan læses: <https://universalis.app/> (Tilgået: 24 April 2024).

*What is WSGI (web server gateway interface) Built In*. Kan læses: <https://builtin.com/data-science/wsgi#> (Tilgået: 11 Maj 2024).

*WSGI*, (2024) Kan læses: <https://wsgi.readthedocs.io/en/latest/index.html> (Tilgået: 15 Maj 2024).

*XIVAPI*. Kan læses: <https://xivapi.com/> (Tilgået: 22 April 2024).

## 16. Bilag

GE tracker screenshots: <https://www.ge-tracker.com/screenshots>

16.1: Tidlig udvikling

The screenshot displays two main sections of the GE Tracker application.

**Craft Calculator Section:**

- Header:** Sapphire Avenue-Toolkit
- Search Bar:** Search marketboard item... (with placeholder Adamantoise)
- Items:**
  - Grade 8 Tincture of Strength:** Spriggan, Last Update: 2024-05-03 12:37:08, Price: 2136
  - Alche-mist:** 1, Price: 1000
  - Grade 5 Strength Alkahest:** 1, Price: 746
  - Earthbreak Aethersand:** 1, Price: 890

**Retainer Details Section:**

**Albino Karakul Horn**

Retainer name	Listing quantity	Price per unit	Listing total
Bigboobsbunny girl	1	593000	593000
Xanara	1	594999	594999
Zerkpink	1	595000	595000

## 16.2: GE Tracker screenshots

The screenshot shows the GE Tracker dashboard with the following key statistics:

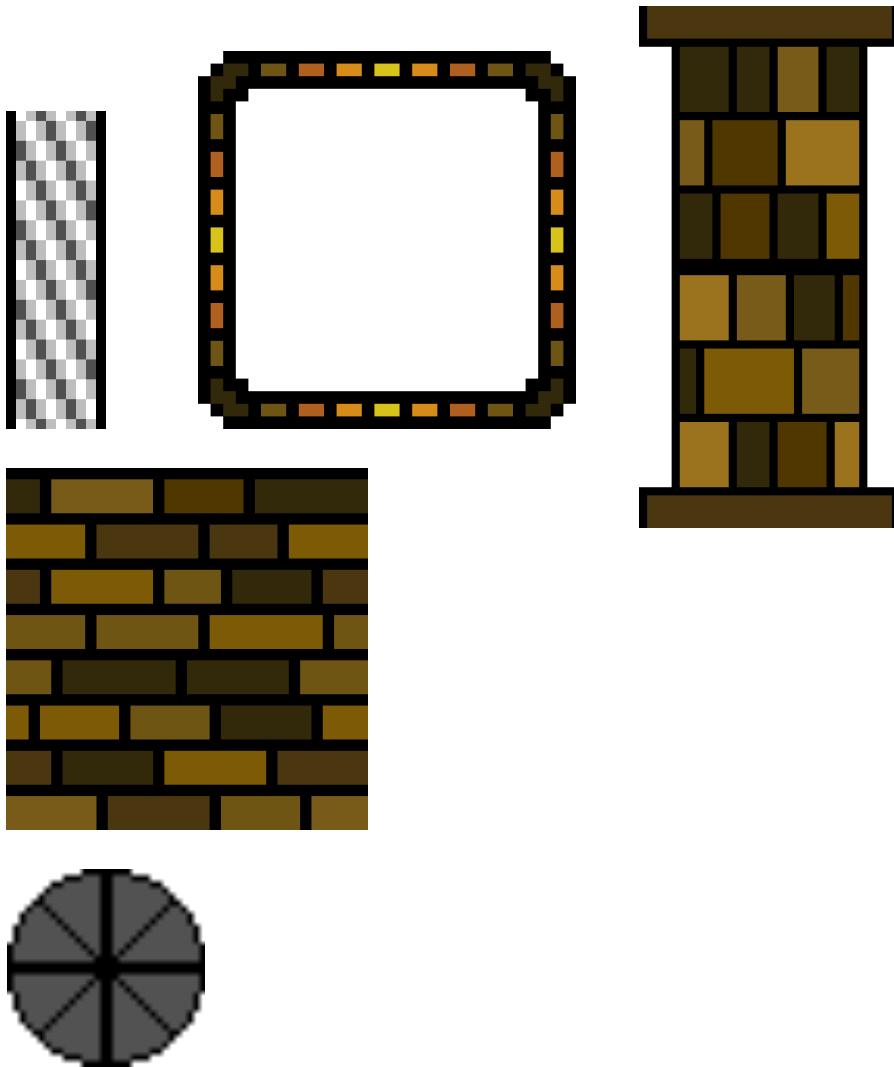
- Total Profit: **143.91m** (143,905,487gp)
- Total Transactions: **381** (**377.71k** average profit)
- Most Profitable Item: **57.57m** (Old school bond)
- Most Traded Item: **238.45k** (Gold amulet (u))
- Most Frequent Item: **46** (Old school bond)
- Active Transactions: **2** (View transactions)

The main area displays a table of favorite items with the following columns: Item, Current Price, Approx. Offer Price, Approx. Sell Price, Approx. Profit (gp), ROI%, Buying Quantity (per hour), and Selling Quantity.

Item	Current Price	Approx. Offer Price	Approx. Sell Price	Approx. Profit (gp)	ROI%	Buying Quantity (per hour)	Selling Quantity
Mystic mud staff	0	37.005 ?	40.509 ?	3.504 ?	9.47%	0	0
Saradomin's tear	4,852,000	4,852,000	4,925,997 ?	73,997 ?	1.53%	0	1
Zenyte amulet	0	12,100,000 ?	12,597,000 ?	497,000 ?	4.11%	0	0
Zenyte bracelet	0	11,701,000 ?	11,999,998 ?	298,998 ?	2.56%	0	0
Zenyte necklace	0	11,850,002 ?	12,100,000 ?	249,998 ?	2.11%	0	0
Abyssal bludgeon	38,944,203	38,805,000 ?	38,944,203	139,203 ?	0.36%	1	0
Dragon warhammer	41,470,000	41,470,000	41,535,000 ?	65,000 ?	0.16%	0	4
Ankou socks	0	3,510,009 ?	3,594,000 ?	83,991 ?	2.39%	0	0
Bandos boots	178,169	178,169	174,499 ?	-3,670 ?	-2.05%	0	1
Bandos godsword	9,492,898	9,483,023	9,498,823	+15,800	0.17%	5	3
Old school bond	6,132,215	6,744,359	6,132,790	-611,569	-9.07%	29	17
Justiciar armour set	46,312,100	46,312,100	46,310,650 ?	-1,450 ?	-0%	0	1

Showing 1 to 17 of 17 entries.

16.3: Artwork



16.5: Final Fantasy XIV - Spil Screenshots



## 16.6: XIVAPI Results

```
{
    "AmountIngredient0": 3,
    "AmountIngredient1": 1,
    "AmountIngredient2": 1,
    "AmountIngredient3": 1,
    "AmountIngredient4": 0,
    "AmountIngredient5": 0,
    "AmountIngredient6": 0,
    "AmountIngredient7": 0,
    "AmountIngredient8": 8,
    "AmountIngredient9": 8,
    "Item": [
        "262445.0"
    ],
    "Quest": {
        "ItemReward0": [
            66656
        ],
        "ItemReward00": [
            66656
        ]
    },
    "RelicItem": {
        "GladiatorItem": [
            0
        ]
    },
    "SpecialShop": {
        "ItemCost00": [
            1769484
        ]
    }
},
"GamePatch": {
    "Banner": "https://i.imgur.com/zv",
    "ExName": "A Realm Reborn",
    "ExVersion": 0,
    "ID": 2,
    "Name": "Patch 2.0: A Realm Reborn",
    "Name_cn": "Patch 2.0: A Realm Reborn",
    "Name_de": "Patch 2.0: A Realm Reborn",
    "Name_en": "Patch 2.0: A Realm Reborn",
    "Name_fr": "Patch 2.0: A Realm Reborn",
    "Name_ja": "\u065b\u0751f\u030a8\u030a",
    "Name_kr": "Patch 2.0: A Realm Reborn",
    "ReleaseDate": 1376611200,
    "Version": "2.0"
},
"GrandCompany": null,
"GrandCompanyTarget": "GrandCompany",
"GrandCompanyTargetID": 0,
"ID": 1675,
"Icon": "\/i\/030000\030446.png",
"IconHD": "\/i\/030000\030446_hri.png",
"IconID": 30446,
"IsAdvancedMeldingPermitted": 0,
"IsCollectable": 0,
"IsCrestWorthy": 0,
"IsDyeable": 0,
"IsGlamourous": 0,
"ItemIngredientRecipe0": [
    {
        "AmountIngredient0": 5,
        "AmountIngredient1": 1,
        "AmountIngredient2": 0,
        "AmountIngredient3": 0,
        "AmountIngredient4": 0,
        "AmountIngredient5": 0,
        "AmountIngredient6": 0,
        "AmountIngredient7": 0,
        "AmountIngredient8": 8,
        "AmountIngredient9": 0,
        "AmountResult": 1,
        "CanHq": 1,
        "CanQuickSynth": 1,
        "ClassJob": {
            "Abbreviation": "BSM",
            "Abbreviation_de": "GRS",
            "Abbreviation_en": "BSM",
            "Abbreviation_fr": "FRG",
            "Abbreviation_ja": "BSM",
            "BattleClassIndex": "-1",
            "CanQueueForDuty": null,
            "ClassJobCategory": 33,
            "ClassJobCategoryTarget": "ClassJobCategory",
            "ClassJobCategoryTargetID": 33,
            "ClassJobParent": null,
            "ClassJobParentTarget": "ClassJob",
            "ClassJobParentTargetID": 9,
            "DohDolJobIndex": 1,
            "ExpArrayIndex": 8,
            "ID": 9,
            "Icon": "\/cj\1\blacksmith.png",
            "IsLimitedJob": null,
            "ItemSoulCrystal": 10338,
            "ItemSoulCrystalTarget": "Item",
            "ItemSoulCrystalTargetID": 10338,
            "ItemStartingWeapon": null,
            "ItemStartingWeaponTarget": "Item",
            "ItemStartingWeaponTargetID": null,
            "JobIndex": null,
            "LimitBreak1": null,
            "LimitBreak1Target": "Action",
            "LimitBreak1TargetID": null,
            "LimitBreak2": null,
            "LimitBreak2Target": "Action",
            "LimitBreak2TargetID": null,
            "LimitBreak3": null,
            "LimitBreak3Target": "Action",
            "LimitBreak3TargetID": null,
            "ModifierDexterity": 90,
            "ModifierHitPoints": 100,
            "ModifierIntelligence": 90,
            "ModifierManaPoints": 100,
            "ModifierMind": 100,
            "ModifierPiety": 100,
            "ModifierStrength": 105,
            "ModifierVitality": 90,
            "MonsterNote": null,
            "MonsterNoteTarget": "MonsterNote",
            "MonsterNoteTargetID": 127,
            "Name": "blacksmith",
            "SkillIndex": 1
        }
    }
]
```