МОНГОЛ УЛСЫН ИХ СУРГУУЛЬ
ХЭРЭГЛЭЭНИЙ ШИНЖЛЭХ УХААН, ИНЖЕНЕРЧЛЭЛИЙН СУРГУУЛЬ
МЭДЭЭЛЭЛ, КОМПЬЮТЕРИЙН УХААНЫ ТЭНХИМ

Анужингийн Сайнзолбоо

АЖИЛ ОЛГОГЧДЫН ӨГӨГДЛИЙН АНАЛИЗ СИСТЕМ ДЭЭР СУУРИЛСАН ЧАТ БОТ

(Chat bot based on sytem analysis of employers' data)

Мэдээллийн технологи (D061303)
Бакалаврын судалгааны ажил

Улаанбаатар

2022 оны 03 сар

МОНГОЛ УЛСЫН ИХ СУРГУУЛЬ
ХЭРЭГЛЭЭНИЙ ШИНЖЛЭХ УХААН, ИНЖЕНЕРЧЛЭЛИЙН СУРГУУЛЬ
МЭДЭЭЛЭЛ, КОМПЬЮТЕРИЙН УХААНЫ ТЭНХИМ

АЖИЛ ОЛГОГЧДЫН ӨГӨГДЛИЙН АНАЛИЗ СИСТЕМ ДЭЭР СУУРИЛСАН ЧАТ БОТ

(Chat bot based on sytem analysis of employers' data)

Мэдээллийн технологи (D061303)
Бакалаврын судалгааны ажил

Удирдагч: _____ Б.Хуягбаатар доктор (Ph.D.)

Гүйцэтгэсэн: _____ А.Сайнзолбоо (18B1NUM1762)

Улаанбаатар

2022 оны 03 сар

Зохиогчийн баталгаа

Миний бие Анужингийн Сайнзолбоо “АЖИЛ ОЛГОГЧДЫН ӨГӨГДЛИЙН АНАЛИЗ СИСТЕМ ДЭЭР СУУРИЛСАН ЧАТ БОТ” сэдэвтэй судалгааны ажлыг гүйцэтгэсэн болохыг зарлаж дараах зүйлсийг баталж байна:

- Ажил нь бүхэлдээ эсвэл ихэнхдээ Монгол Улсын Их Сургуулийн зэрэг горилохоор дэвшүүлсэн болно.
- Энэ ажлын аль нэг хэсгийг эсвэл бүхлээр нь ямар нэг их, дээд сургуулийн зэрэг горилохоор оруулж байгаагүй.
- Бусдын хийсэн ажлаас хуулбарлаагүй, ашигласан бол ишлэл, зүүлт хийсэн.
- Ажлыг би өөрөө (хамтарч) хийсэн ба миний хийсэн ажил, үзүүлсэн дэмжлэгийг дипломын ажилд тодорхой тусгасан.
- Ажилд тусалсан бүх эх сурвалжид талархаж байна.

Гарын үсэг: _____

Огноо: _____

ГАРЧИГ

ЗУРГИЙН ЖАГСААЛТ

ХҮСНЭГТИЙН ЖАГСААЛТ

Кодын жагсаалт

УДИРТГАЛ

Мэдээллийн технологи эрчимтэй хөгжиж буй өнөөгийн нийгэмд байгууллага үйл ажиллагаа явуулж эхэлсэн цагаасаа эхлэн өгөгдлийг үйлдвэрлэсээр байдаг. Тэдгээр өгөгдлийг байнга хадгалах нь өгөгдлийн сангийн нөөцөд хортой байдаг тул өгөгдөлд шинжилгээ хийж, тэдгээрээс шаардлагатай өгөгдлүүдийг түүвэрлэн хадгалах нь чухал юм.

Өнөөдөр бид дэлхий нийтээрээ хурдтай амьдралын хэмнэлд ажиллаж, амьдарч байна. Мөн зах зээлийн хөгжил, ажил олгогчийн эрэлт хэрэгцээ ажил хайгчийн хүсэл онирхлыг оновчтой бөгөөд хурдан холбож өгөх нь нэн шаардлагатай. Өнөөгийн байдлаар энэ эрэлт хэрэгцээг хангасан тодорхой шийдвэрлэсэн мэдээллийн систем хомс байна. Иймд энэхүү бакалаврын судалгааны ажлаар ажил олгогч болон ажил идэвхтэй хайгч хоёрыг түргэн шуурхай холбож өгөх чатбот системийг хөгжүүлж байна.

1. СЭДВИЙН ТАНИЛЦУУЛГА

1.1 Оршил

Энэхүү бакалаврын судалгааны ажлын хүрээнд “Ажил олгогчдын өгөгдлийн анализ систем дээр суурилсан чатбот” сэдвийн дагуу ажил хайгчдыг ажлын байрны мэдээллээр хангах чатбот системийг хөгжүүлнэ. Ажлын байрны мэдээллийг Data Scraping аргын тусламжтайгаар, системд шаардлагатай мэдээллийг өгөгдлийн сангийн хэлбэрт оруулан бүтэцтэйгээр нэгтгэн авах бөгөөд үүнээс ажил хайгчдын дунд байдаг түгээмэл асуултуудын хариултыг өгнө.
Мөн энэ системд машин сургалтын арга болох Language Understanding-ийг ашиглан хэрэглэгчийн асуултыг таамаглаж оновчтой хариулт өгөх боломжийг олгох юм.

1.2 Зорилго

Ажлын хайгчдын хэрэгцээт асуултад хариулж, ажлын байрны хүртээмжийг нэмэгдүүлэхэд энэхүү системийн гол зорилго оршино.

1.3 Зорилт

Дээрх зорилгод хүрэхийн тулд дараах зорилтуудыг тавьсан. Үүнд:

- Ашиглагдах технологиудыг сонгох, судлах
- Ижил төстэй системийн судалгаа хийх
- Системийн шинжилгээ хийх
- Системийг зохиомжлох
- Системийг хөгжүүлэх, сайжруулалт хийх

1.4 Алсын хараа

Ажлын байрны дэлгэрэнгүй мэдээллийг цуглуулснаар цаашид тэдгээрт шинжилгээ хийж хамгийн их эрэлттэй, өндөр цалинтай ажлын байр гэх зэрэг мэдээллүүдийг систем хэрэглэгчдэд хүргэх боломжтой юм.

2. СИСТЕМИЙН СУДАЛГАА

2.1 Системийн судалгаа

Сонгосон сэдэв болох “Ажил олгогчдын өгөгдлийн анализ систем дээр суурилсан чатбот” сэдвийн хүрээнд судалгаа хийхдээ чатбот системийн талаар болон өгөгдөл цуглуулгын аргын талаар судалсан. Үүний дараа ижил төстэй системийн болон ашиглагдах технологийн талаар судалгааг хийсэн болно.

2.1.1 Чатбот систем

Чатбот систем нь ихэвчлэн хэрэглэгчийн асуултыг хиймэл оюун ухааны тусламжтайгаар ойлгож, хариултыг автоматжуулах үндсэн зорилготой компьютерийн програм хангамж юм. Орчин үед хэрэглэгчдэд туслах үндсэн үүргийн дагуу чатбот системийг байгууллагууд олон янзаар ашиглах болсон. Тэдгээрээс дурдвал,

- Цэс дээр суурилсан чатбот (Menu-based chatbot)
- Түлхүүр үгийг танихад суурилсан чатбот (Keyword recognition-based chatbot)
- Машин сургалтын чатбот (Machine learning chatbot)

Цэс дээр суурилсан чатбот

Өнөөгийн зах зээлд хэрэгжиж буй чатботуудын хамгийн энгийн бөгөөд түгээмэл хэлбэр юм.[1] [1] Хэрэглэгчийн асууж болох асуултуудыг урьдаас таамаглан хариултуудыг мод хэлбэртэйгээр бүтэцлэн хадгалдаг. Хэрэглэгч хүссэн хариултаа авахын тулд системийн хадгалсан хариултаар аялах хэрэгтэй болдог. Бусад чатботтой харьцуулбал, хариулт хязгаарлагдмал бөгөөд хэрэглэгчээс олон асуулт асууж цаг их шаарддагаараа сул талтай байдаг.

[1] `https://www.engati.com/blog/types-of-chatbots-and-their-applications`

Түлхүүр үгийг танихад суурилсан чатбот

Энэхүү чатбот нь хэрэглэгчийн бичсэнийг уншиж тохиромжтой хариултыг өгдөг. Ингэхдээ өгүүлбэрийг хиймэл оюун ухааны нэг хэсэг болох эх хэлний боловсруулалт (Natural Language Processing)-ын тусламжтайгаар шинжилж түлхүүр үгийг таньж хариултыг өгдөг. Ижил төстэй олон асуултад хариулах эсвэл түлхүүр үг дутуу үед амжилтгүй болдог. Мөн хэрэглэгч хүссэн хариултаа олж чадахгүй байх болон үр дүн муутай хариулт өгсөн тохиолдолд цэс дээр суурилсан чатботыг хослуулан ашиглах нь найдвартай болдог бөгөөд түгээмэл шийдлүүдийн нэг байдаг.

Машин сургалтын чатбот

Энэ төрлийн чатбот нь өмнө хэрэглэгчийн харилцан яриан дээр хиймэл оюун ухаан болон машин сургалтын тусламжтайгаар шинжилгээ хийж, хэрэглэгчийн зан төлөв, асуултын хэв маягаас суралцдаг. Ингэснээрээ чатботод хэрэглэгчийн зарцуулах цаг эрчимтэйгээр буурах буюу хариултаа авах алхам багасгах ба хэрэглэгчийн туршлага (UX) нь түүнийгээ даган өсөх нь энэхүү чатботын үндсэн зорилго болно.

Чатботыг сонгох

Машин сургалтын чатбот нь илүү уян хатан хэрэглэгчдэд ээлтэй чатботыг бий болгодог боловч хөгжүүлэхэд цаг хугацаа их шаардагдах ба машин өөрөө суралцахад мөн хугацаа шаардагддаг. Иймд системийн нөөц, шаардлагыг харгалзан үзэж энэхүү судалгааны ажлаар түлхүүр үг танихад суурилсан чатботыг хэрэгжүүлэхийг зориод байна.

2.1.2 Өгөгдөл цуглуулгын арга

Өгөгдөл цуглуулах (data scraping) нь хэрэглэгчдэд харагдаж буй өгөгдлийг олон янзын сувгаас цуглуулан хувийн орчинд хадгалан цаашид ашиглах боломжийг олгодог хамгийн үр дүнтэй автомат өгөгдөл олборлох арга юм. Ихэвчлэн өгөгдөл цуглуулах арга нь вэбсайтаас шаардлагатай өгөгдлийг цуглуулахад ашигладаг. Өгөгдөл цуглуулж буй хүнээс хамааран олборлосон өгөгдлийг таслалаар тусгаарлагдсан утгын (Comma-Separated Values) файл эсвэл

өгөгдлийн санд хадгалах боломжтой бөгөөд нэгэнт цуглуулсан их хэмжээний өгөгдөлд судалгаа шинжилгээ хийх, худалдаа, борлуулалтын хэрэгсэл болгох зэрэг олон төрлийн боломжийг олгодог.

Вебсайтаас өгөгдлийг олборлох хамгийн түгээмэл арга нь HTML parsing буюу HTML-ийг задлан шинжлэх юм. Энэ нь вебсайтын HTML болох сайтын үндсэн бүтцийг агуулгынх нь хамтаар хуулах бөгөөд авах гэж буй өгөгдлийн зан төрхийг нь зааж өгснөөр доторх агуулгыг хамгийн хялбар бөгөөд автомат байдлаар цуглуулдаг юм. Цуглуулга хийх 2 үндсэн арга байдаг. Үүнд:

- Өгөгдлийг цуглуулж, задлах (Data scraping)
- Өгөгдлийг олж илрүүлж, хаягийг цуглуулах (Data crawling)

Өгөгдлийг цуглуулж, задлах

Нэг үгээр хэлбэл өгөгдлийг цуглуулж, задлах нь зааж өгсөн хаягийн дагуу шаардлагатай өгөгдлийг задалж, хэрэгтэй агуулгыг хөгжүүлэгчдэд өгдөг бөгөөд хүссэн өгөгдлөө задлан авах боломжийг олгодгоороо давуу талтай. Өөрөөр хэлбэл өгөгдөл олборлох програм нь зорилго буюу даалгавраа мэдэж байгаа юм.

Өгөгдлийг олж илрүүлж, хаягийг цуглуулах

Энэхүү аргачлал нь хаяг тодорхой бус үед түүнийг олж илрүүлж шаардлагын дагуу хаягийг, зарим тохиолдолд өгөгдлийг цуглуулдаг. Системийн шаардлагын дагуу өгөгдлийг цуглуулах үед хаяг алгасах, дутуу өгөгдөл цуглуулахаас сэргийлдэг давуу талтай.

Ихэвчлэн энэхүү хоёр аргыг хослуулан ашигладаг бөгөөд шаардлагад нийцэх өгөгдлийг үлдээлгүй бүгдийг нь олоход *data crawling*-ийг ашиглах бол олсон өгөгдлийг задалж, шинжлэн өгөгдлийн санд хадгалах үйлдлийг *data scraping* хийдэг. Жишээлбэл, худалдааны сайтын бараа бүтээгдэхүүний өгөгдлийг цуглуулах гэж байгаа гэж үзвэл, барааны ангиллын хаягуудыг өөрчлөгдөх бүрд хадгалан өгөгдлийг цуглуулна. Өөрөөр хэлбэл нэг нь өөрчлөлт гарахыг ажиглаж вебсайтаар мөлхөж байх бол нөгөө нь шаардлагын дагуу бүх хэрэгтэй өгөгдлийг хэдийн цуглуулсан

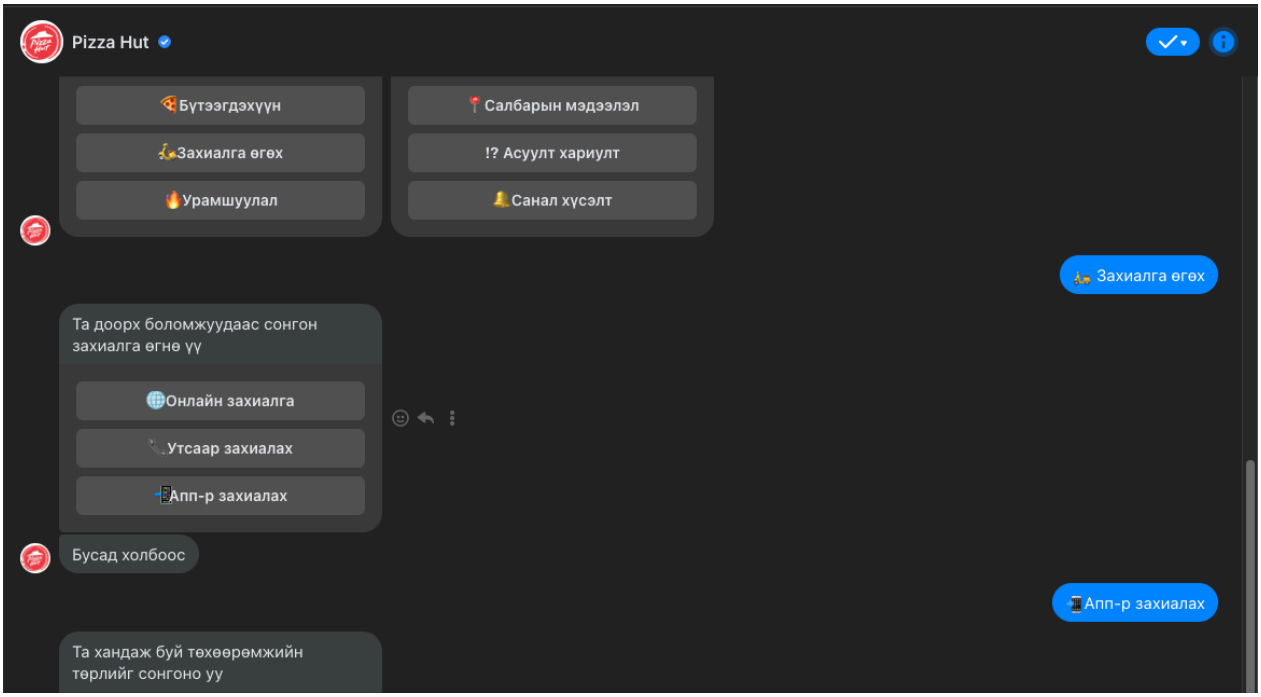
байна. Энэхүү бакалаврын судалгааны ажлын хүрээнд өгөгдлийг CSV файл үүсгэн хадгалж цаашид ашигласан болно.

2.2 Ижил төстэй системүүд

Гадаад ба дотоодын байгууллагуудын үйл ажиллагаандаа хэрэгжүүлдэг чатбот системүүдээс, *Domino's Pizza & Pizza Hut* болон WHO's Chat bot гэсэн гурван чатботыг сонгон авч судалгаанд оруулав.

2.2.1 Domino's Pizza & Pizza Hut

Domino's pizza хоолны газар нь захиалгын алхмаас эхлээд бүх мэдээллийг ганцхан *Facebook messenger chatbot* хангадаг. Чатбот эрчээ авч эхэлсэн шалтгаан нь хүмүүс, бусад хүмүүсийг хүлээлгүйгээр үйлчилгээ авах, тусламж авах зэрэг үйлчилгээг зэрэг нэвтрүүлсэнтэй холбоотой билээ. Үүний нэгэн адилаар Монголд үйл ажиллагаа явуулж буй Pizza Hut Mongolia юм.

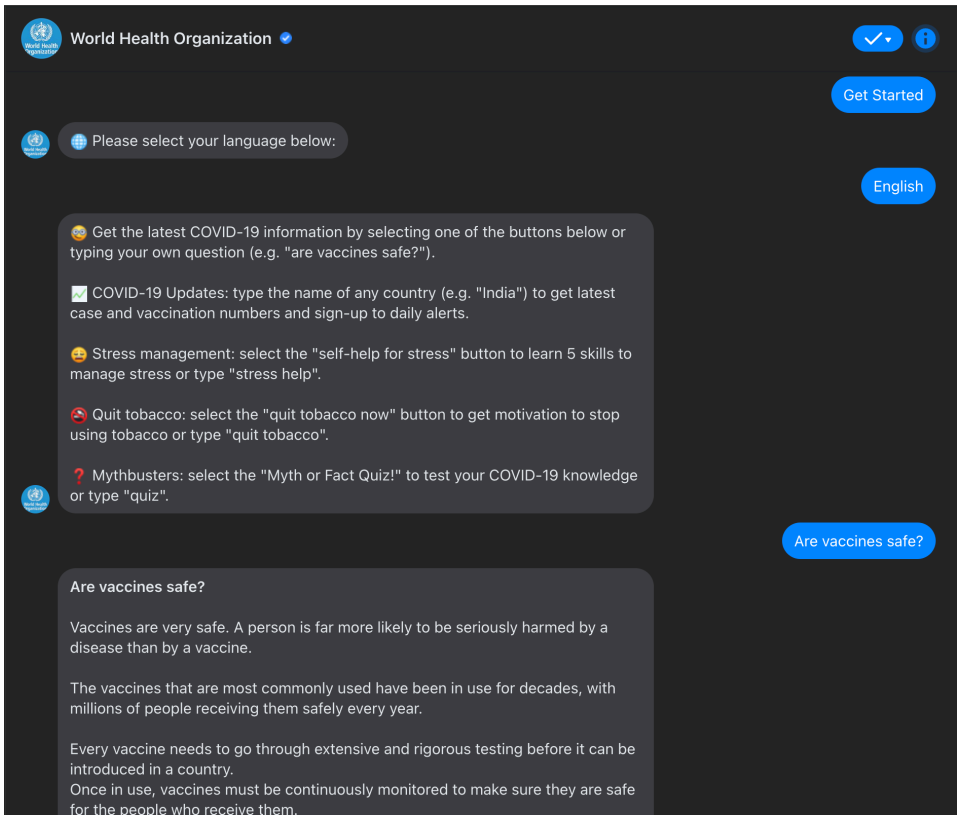


Зураг 2.1: Pizza Hut chat bot

Үйлчлүүлэгчдийн захиалга хүлээх хугацааг багасгахын тулд захиалгын үйл явцыг хурдасгаснаар тодорхой хэмжээнд нөлөөлж байгаа нь дээрх 2 жишээнээс харагдаж байна.

2.2.2 World Health Orgazination's Chat bot

Цар тахал болох коронавирусын эрчимтэй тархаж байх үед дэлхийн өнцөг булан бүрд оршин суугаа хүмүүст цар тахлын мэдээлэл, урьдчилан сэргийлэх арга, баталгаатай эх сурвалжийн мэдээллээр хангах зорилготой чатбот юм. Дэлхий нийтээр вакцинжуулалтын хөдөлгөөн өрнөж байх үеэр вакцины талаарх мэдээлэл, архаг хууч өвчинд нөлөөлөх талаар найдвартай, хамгийн сүүлийн үеийн албан ёсны мэдээллийг өгдөг. Хэдий халдварын тоо буурч, нийгэм өөрөө дасан зохицож байгаа хэдий ч Дэлхийн Эрүүл Мэндийн байгууллага үүргээ гүйцэтгэж чухал эх сурвалжаар хангасаар байгаагийн шинж юм.



Зураг 2.2: WHO's chat bot

2.3 Технологийн судалгаа

АЖИЛ ОЛГОГЧДЫН ӨГӨГДЛИЙН АНАЛИЗ СИСТЕМ ДЭЭР СУУРИЛСАН ЧАТ БОТ-ыг хөгжүүлэхдээ өгөгдөл цуглуулгыг *python* хэлний сан болох *BeautifulSoup* HTML өгөгдөл задлах технологийг ашигласан бөгөөд чатбот системийн түлхүүр үг таних технологийг *Python* хэлний *Framework* болох *SentenceTransformers*-ийг сонгон хөгжүүлэлтийг хийсэн. Харин цуглуулсан өгөгдлийг *CSV* файлд хадгалан, *Microsoft Bot Framework*-ийг чатбот хөгжүүлэлтэд ашиглан судалгааг дараах байдлаар хийсэн болно.

2.3.1 Python

Python нь дээд түвшний маш олон төрлийн програмчлалыг өөртөө шингээсэн хэл юм. Хэлний сан болон *framework*-үүд нь тасралтгүй сайжирч, шинэчлэгдэж байдаг тул бүх л төрлийн програмчлалын аргуудыг гүйцэтгэж болдог. Орчин үед машин сургалт, хиймэл оюун ухаан болон эх хэлний боловсруулалтад(NLP) түгээмэл ашигладаг болсон бөгөөд веб хүртэл хийх боломжтойгоороо давуу талтай юм. Үүнээс гадна анхлан суралцаж буй хүмүүст ойлгоход хялбар *syntax*-ийн дүрэмтэй байдаг тул хэрэглэгчдийн тоо нь javascript, java хэлүүдтэй өрсөлдөхүйц байдаг.

Зураг 2.3: Python лого

```
1 x = 5
2 name = 'Sainzolboo'
3 print(x)
4 print(name)
```

Код 2.1: Python энгийн жишээ

Python програмчлалын хэл нь ойлгоход маш хялбар бөгөөд өөр дээрх функцууд нь шууд утгаараа ойлгомжтой байдаг. Syntax-ийн хувьд ; ашигладаггүй ба догол мөрөөр програмчлалын үндсэн схемийг гаргадгаараа онцлог хэл юм.

2.3.2 BeautifulSoup

Өгөгдөл цуглуулгын олон технологиудын нэг нь *BeautifulSoup* бөгөөд *python* програмчлалын хэлний сан юм. Энэ нь өгсөн вебсайтын хаяг (Url)-ийн дагуу бүх HTML өгөгдлийг агуулгын хамтаар нь хэрэглэгчид өгдөг. HTML хэл нь мод хэлбэртэй байдаг бөгөөд түүний хүүхэд элементүүдийн агуулгыг шаардлага болон түлхүүр үгийн дагуу цуглуулах зарчмаар ажилладаг.

Зураг 2.4: BeautifulSoup лого

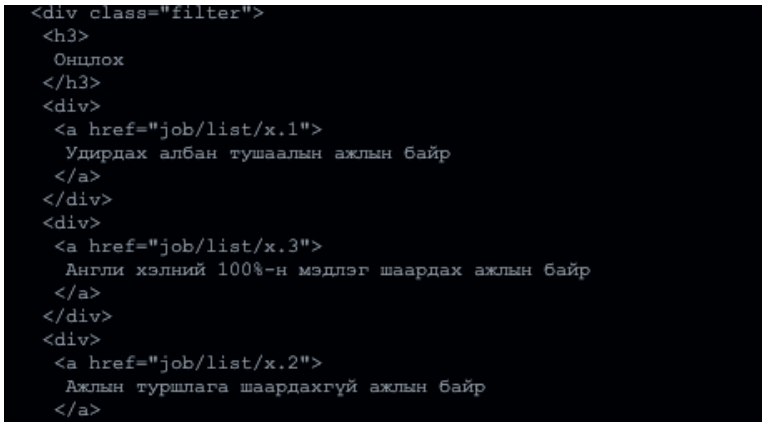
Бакалаврын судалгааны ажлын сэдвийн дагуу ажлын байр олгогчдын мэдээлэл болон ажлын байрны мэдээллийг **zangia.mn**-ээс *BeautifulSoup* ашиглан цуглуулсан. Доорх кодын жишээнд бүх ажлын байрны ангилал болон шүүлтүүрийн агуулгыг цуглуулсан бөгөөд жишээнд зориулж зөвхөн эхний ангиллын мэдээллийг харуулав.

```
from bs4 import BeautifulSoup
import requests
from urllib.error import HTTPError

url = 'https://zangia.mn/'
```

```
try:
    response = requests.get(url)
    response.raise_for_status()
except HTTPError as error:
    print(error)
soup = BeautifulSoup(response.text, 'html.parser')
navigatorList = soup.find_all('div', class_='filter')
print(navigatorList[0])
```

Код 2.2: BeautifulSoup жишээ өгөгдөл цугуулалт

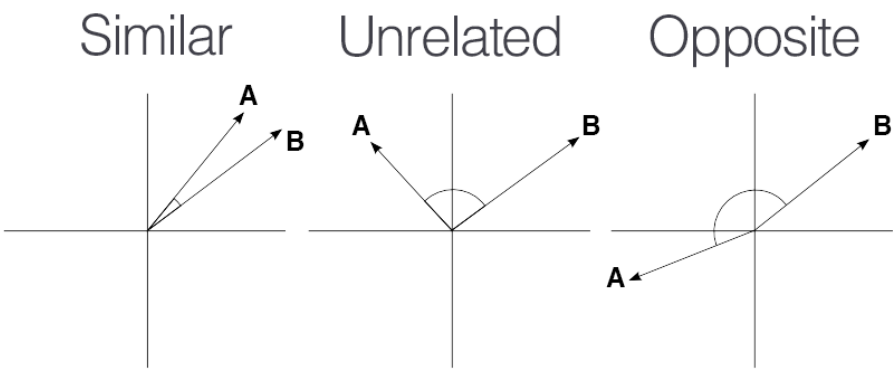


Зураг 2.5: Өгөгдөл цуглуулалтын жишээний үр дүн

2.3.3 SentenceTransformers

Python хэлний framework болох SentenceTransformers [2] буюу өгүүлбэр хувиргалт нь өгүүлбэр болон текстийн ижил төстэй байдал болон утгын хувьд адил байдлыг *cosine-similarity*[2] -ийн тусламжтайгаар тооцоолдог. Энэхүү тооцооллыг цаашид өгүүлбэрийн ижил төстэй байдлыг харьцуулах, хайлт хийх, түүнд шинжилгээ хийх зэргээр ашиглаж болно. Доорх зурагт өгүүлбэрт хувиргалт хийж, шинжилгээний үр дүнгийн вектор хоорондын өнцгөөр хэрхэн тодорхойлогддог болох талаар харуулав.

[2]Cosine-similarity нь өгөгдлийн шинжилгээнд 2 тооны ижил төстэй байдлыг вектор үржвэрээр илэрхийлдэг.



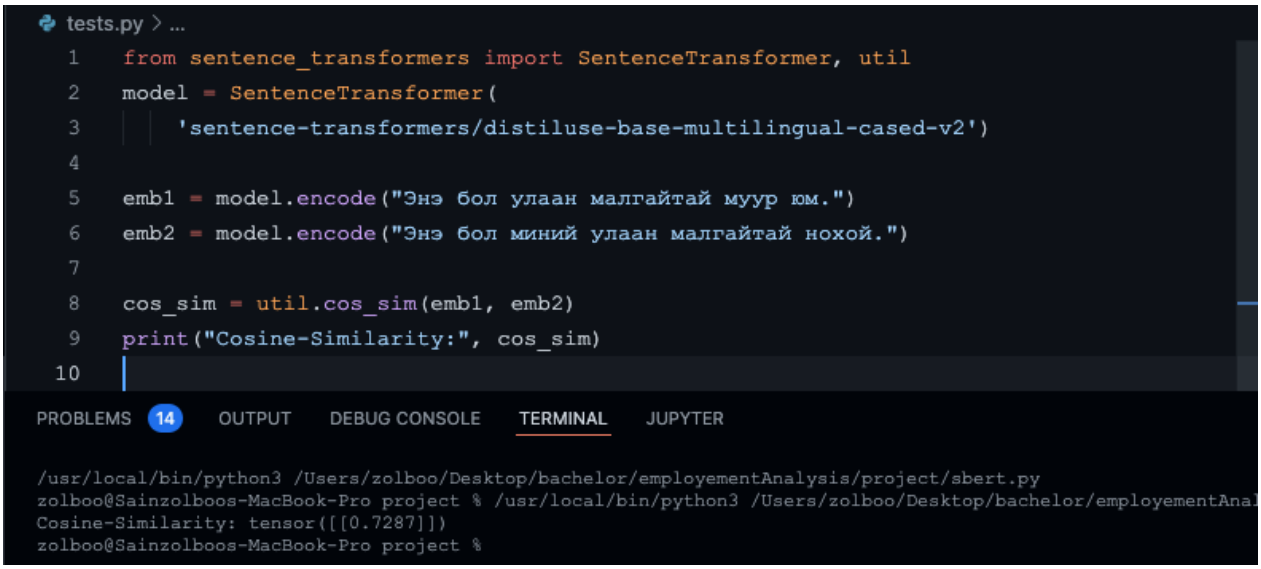
Зураг 2.6: Cosine similarity утгын график

SentenceTransformers-ийг дэлхийн 100 гаруй хэл дээр урьдчилан бэлтгэн, сургасан эх хэлний боловсруулалт (NLP)-ын загваруудыг ашиглаж болдгоороо давуу талтай.

Зураг 2.7: SBERT.net лого

Чатбот системийн хувьд монгол хэлийг танин ашиглах боломжтой загвар болох *distiluse-base-multilingual-cased-v2*[3]-ийг ашиглан хийж гүйцэтгэв.

Хоёр өгүүлбэрийг *cosine-similarity* ашиглан ижил төстэй байдлыг илэрхийлэх жишээг доор харууллаа. Эх кодыг utf-8 формат танихгүй байсан тул зураг хэлбэрээр орууллаа.



Зураг 2.8: cosine-similarity ашигласан жишээ

2.3.4 Linear Regression - Шугаман Регресс

Linear Regression буюу шугамар регрессийн загвар нь шулуун шугамыг ашигладаг бол логик болон шугаман бус регрессийн загвар нь муруй шугамыг ашигладаг. Шугаман регресс нь бие даасан хувьсагч хэрхэн өөрчлөгдөхийг тооцоолох боломжийг олгодог. Хоёр тоон хувьсагчийн хоорондын хамаарлыг тооцоолоход шугаман регрессийг ашигладаг бөгөөд ихэвчлэн дараах нөхцөлд ашиглагддаг. Үүнд:

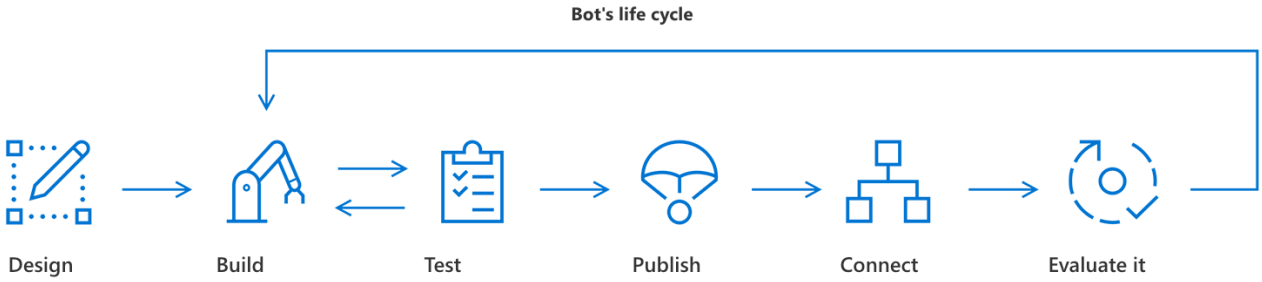
- Хоёр хувьсагчийн хоорондын нөлөөлөл
- Бие даасан хувьсагчийн тодорхой утга дахь хамааралтай хувьсагчийн утга
- Вариацын нэгэн төрлийн байдал
- Ажиглалтын бие даасан байдал
- Нормал байдал

2.3.5 Comma Separated Values - CSV файл

CSV нь өгөгдлийн утгуудыг тусгаарлахад таслал ашигладаг текст файл юм. Файлын мөр бүр нь өгөгдөл байдаг бөгөөд харгалзах утгуудад текст файлыг бичих энгийн өгөгдөл хадгалах технологи юм. Их өгөгдөлтэй хялбар харьцах боломжийг олгодгоороо давуу талтай.

2.3.6 Microsoft Bot Framework

Bot Framework[3] нь Microsoft-ийн *Azure Bot Service*-ийн тусламжтайгаар чатботыг турших, үүсгэх, удирдах, хэрэглээнд нэвтрүүлэх гэх мэт боломжуудыг нэг дор хангаж өгдөг. Энэхүү боломжуудын хүрээнд асуулт хариултыг зохицуулах, хэрэглэгчид зориулсан User Interface бүтээх, Language Understanding аргыг ашиглах гэх мэт үйлдлүүдийг хийх боломжтой.
Bot бүтээх үйл явцыг Azure Bot Service болон Bot Framework нь ихэд хөнгөвчилж өгдөг бөгөөд доорх зурагт үзүүлсэн дарааллын дагуу Bot системийг бүтээдэг.



Зураг 2.9: Чатботын амьралын мөчлөг

Design

Design буюу загварчлах нь төслийн төлөвлөгөөг гаргах юм. Өөрөөр хэлбэл, системийн зорилго, үйл явц, хэрэглэгчийн хэрэгцээг сайтар судлах нь амжилттай Bot систем бүтээх чухал хэсэг юм.

Build

Бот системийг угсрах буюу хөгжүүлэх үйл явц юм. Энэ алхамд хөгжүүлэгч хэрэглэгчийн

[3]https://dev.botframework.com/

харагдах хэсгийг загварчлах бөгөөд хөгжүүлэлтийн орчин нь *Azure Portal*, JavaScript, Python болон C програмчлалын хэлнүүдээс сонгож хөгжүүлэлтийг гүйцэтгэх явц юм. Мөн системийн шаардлагыг тодорхойлсны дагуу бот системийг өргөжүүлж ашиглах боломжтой бөгөөд тэдгээрээс дурдвал:

- Эх хэлний боловсруулалт (NLP)
- Асуулт хариултыг сайжруулан мэдлэгийн сан үүсгэх
- Хэрэглэгчийн интерфейсийг сайжруулах

Test

Програм хангамжийн хөгжүүлэлтийн амьдралын мөчлөгийн адилаар тестийн үйл явцыг алгасаж болохгүй. Нэгэнт хэрэглэгчийн гарт бот системийг оруулахаас өмнө гарч болох алдаа дутагдлыг засан сайжруулах шаардлагатай. Иймд Bot системийг publish хийхээс өмнө заавал туршиж үзэх шаардлагатай. Энд Microsoft-ийн өөрсдийнх нь бие даасан програм болох *Bot emulator*-ийг ашиглан хөгжүүлэлтийн орчинд туршиж үзэх боломжийг олгодог.

Publish

Тестийн шатны дараа бот систем ашиглахад бэлэн болсон гэж үзсэн үед төсөл эсвэл чатботыг олон нийтэд ил болгох явдал юм.

Connect

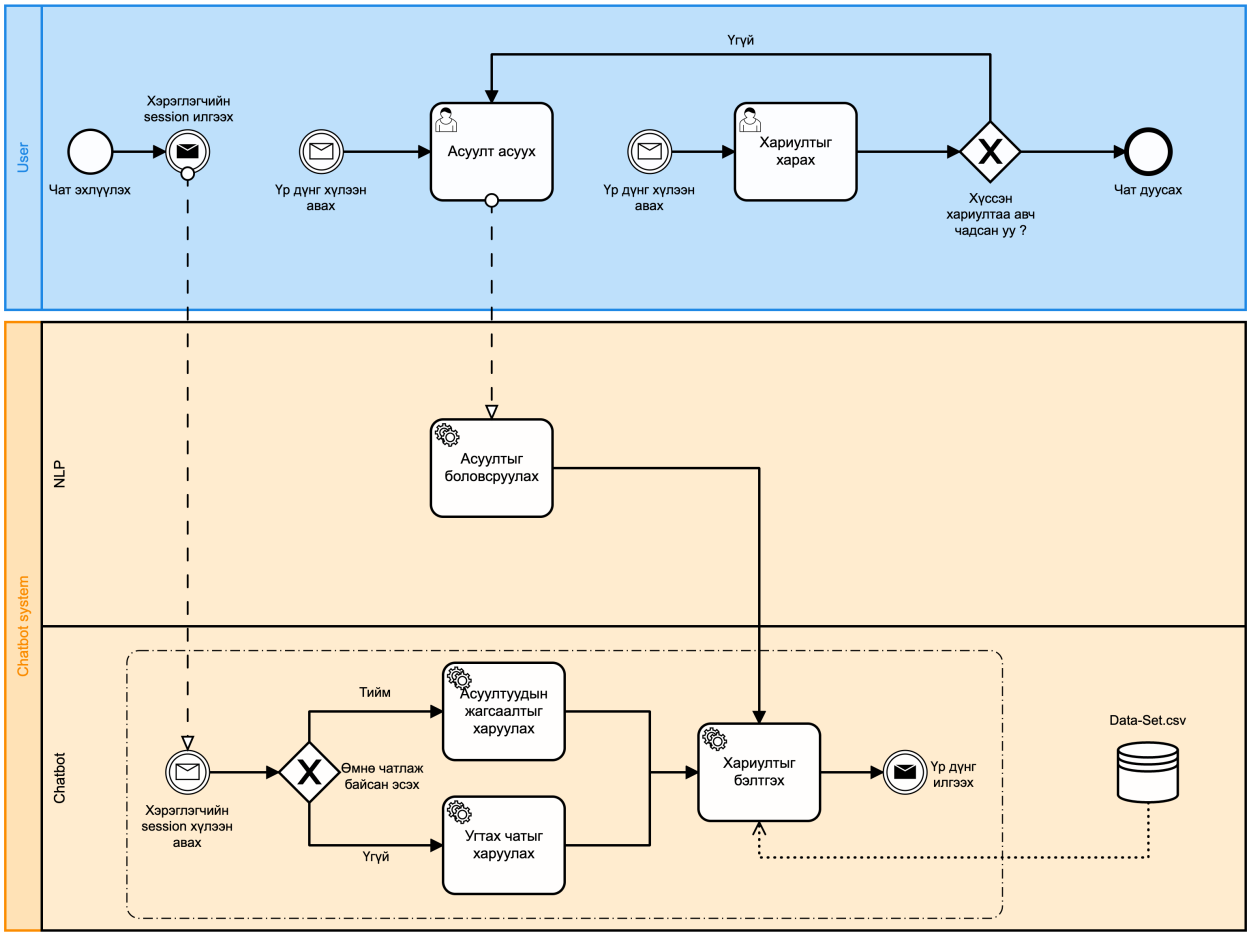
Bot системээ Facebook messenger, Microsoft Teams, Telegram, Skype гэх мэт чат сувгуудыг өөрийн Bot-той холбоно.

Ингэж бүх мөчлөгийг дууссаны дараа хөгжүүлэгч хэрэглэгчийн ашиглаж буй байдал дээр анализ хийж системийг дахин сайжруулах боломжтой бөгөөд буцаад угсрах үйл явцруу шилжих юм.

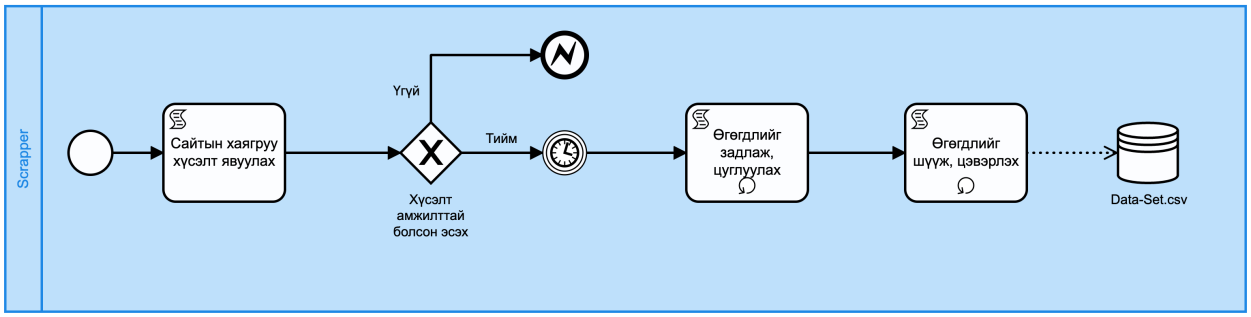
3. СИСТЕМИЙН ШИНЖИЛГЭЭ

3.1 Бизнесийн үйл ажиллагааны шинжилгээ

Бизнес процессийн модель нь чатбот системийн үндсэн процесс буюу үйл ажиллагааны явцыг BMPN-2.0 ашиглан дүрслэн харуулав [5]. Диаграммд дүрслэхдээ оролцогч талууд болох системүүдийг тус тусын *pool* дотор дүрсэлсэн бол дэд процесс буюу *subprocess*-ийг *lane*-д дүрсэлж хоорондын хамаарлыг харууллаа.



Зураг 3.1: BPMN-1



Зураг 3.2: BPMN-2

3.2 Хэрэглэгч

Чатбот системийг ямар ч хүн хэрэглэх боломжтой бөгөөд олон нийтэд нээлттэй байна. Системийн гол зорилго нь ажил хайж буй хэрэглэгчдэд ажлын байрны цогц мэдээллийг олгох зорилготой байх тул хэрэглэгчдийг дараах байдлаар тодорхойлж болно. Үүнд:

- Ажлын байр хайж буй хүн
- Хөгжүүлэгч

3.3 Функционал шаардлага

Дараах хэсэгт чатбот системд тавигдах функционал шаардлагуудыг харуулсан болно.

ФШ 1 Чатбот нь харилцан яриа эхэлмэгц хариу өгдөг байна.

ФШ 2 Чатбот нь ямар ч оролтод хариу өгнө.

ФШ 3 Хэрэв чатбот нь оролтод хариу өгч чадхааргүй байвал бусад асуултуудыг санал болгож ойлгомжгүй утга оруулсныг илэрхийлнэ.

ФШ 4 Чатботын санал болгох асуултууд нь цэс хэлбэртэй харагдана.

ФШ 5 Чатботын цэсэн дээр нэг товшилтоор асуултын хариултыг харуулдаг байна.

ФШ 6 Алхам бүрд үндсэн цэсрүү буцах сонголтыг харуулдаг байна.

ФШ 7 Чатботны хариулт нь текстэн хэлбэрээр хэрэглэгчид харагдана.

ФШ 8 Чатбот нь зөвхөн Монголоор бичсэн асуултад хариулт өгнө.

ФШ 9 Чатбот нь дэлгэрэнгүй мэдээллийг цэс хэлбэрээр сонгуулан харуулж чаддаг байна.

3.4 Функционал бус шаардлага

Бэлэн болон найдвартай байдал (Availability & Reliability)

ФБШ 01 Чатбот систем өдрийн аль ч цагт 99.999% ажиллагаатай байх ёстой.

ФБШ 02 Ямар ч хүсэлт ирсэн чатбот 100% хариу өгдөг байна.

Гүйцэтгэлтэй байдал (Performance)

ФБШ 03 Чатботын байршуулсан сувагт, шаардлагаас хамаарч ямар ч төхөөрөмжөөс хандаж болно.

ФБШ 04 Зарим тохиолдолд чатботын гүйцэтгэл нь хэрэглэгчийн интернет болон төхөөрөмжийн үйлдлийн системийн хувилбараас хамаарч болно.

Дэмжих чадвар (Supportability)

ФБШ 05 Чатботын эх кодыг *github* дээр нээлттэй эхийн систем хэлбэрээр байршуулна.

Хэрэгцээт байдал (Usability)

ФБШ 06 Чатбот нь хэрэглэхэд хялбар, ойлгомжтой байна.

ФБШ 07 Чатботны цэс нь ойлгомжтой цөөн үгээр илэрхийлэгдсэн байна.

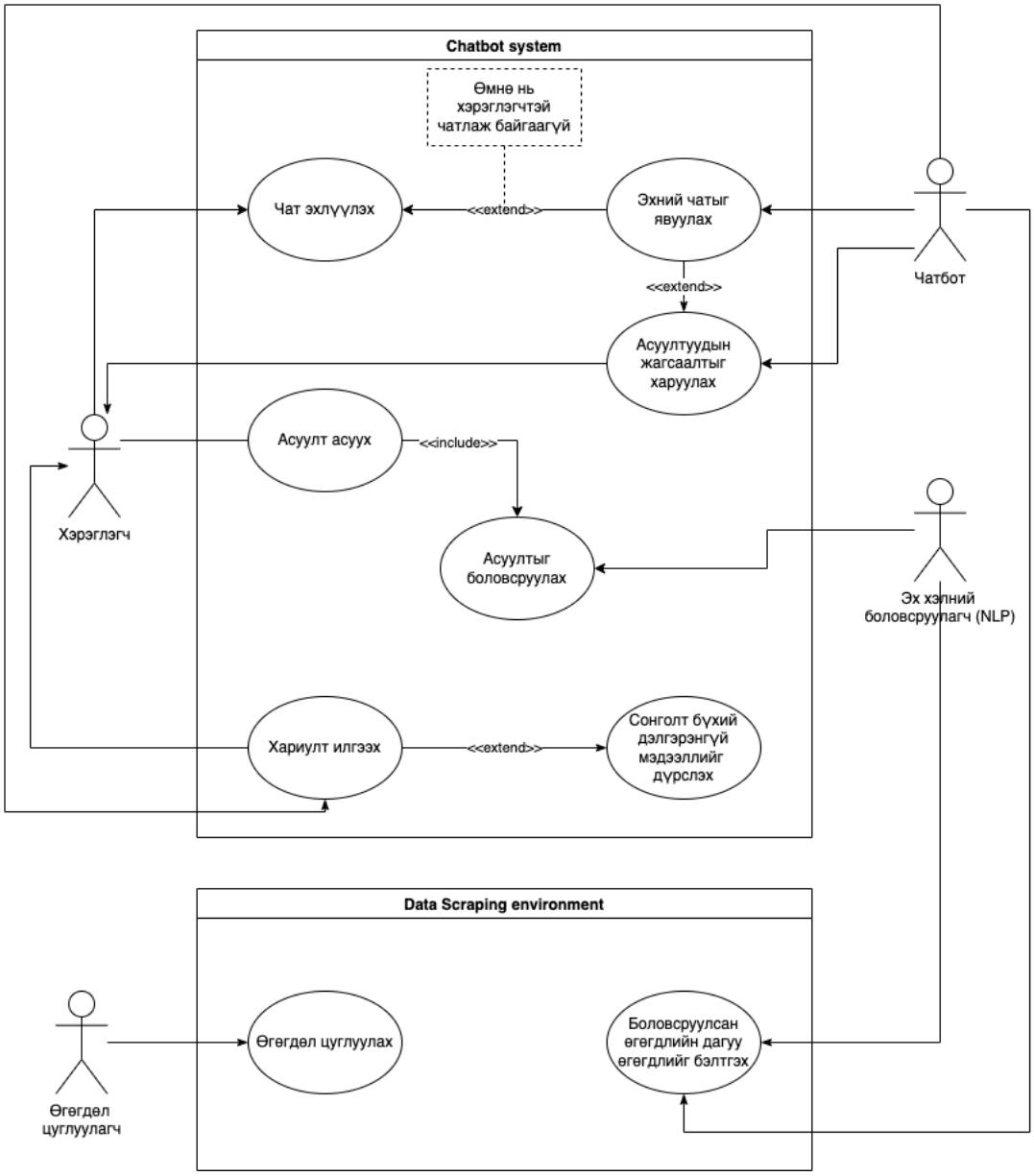
ФБШ 08 Чатботны цэсийн хэмжээ дарагдахуйц том байна.

Аюулгүй байдал(Security)

ФБШ 09 Чатбот системийн байршуулсан сувгийн стандартын дагуу хэрэглэгчийн мэдээллийг өгөгдлийн санд хадгалахгүй байна.

3.5 Use case диаграмм

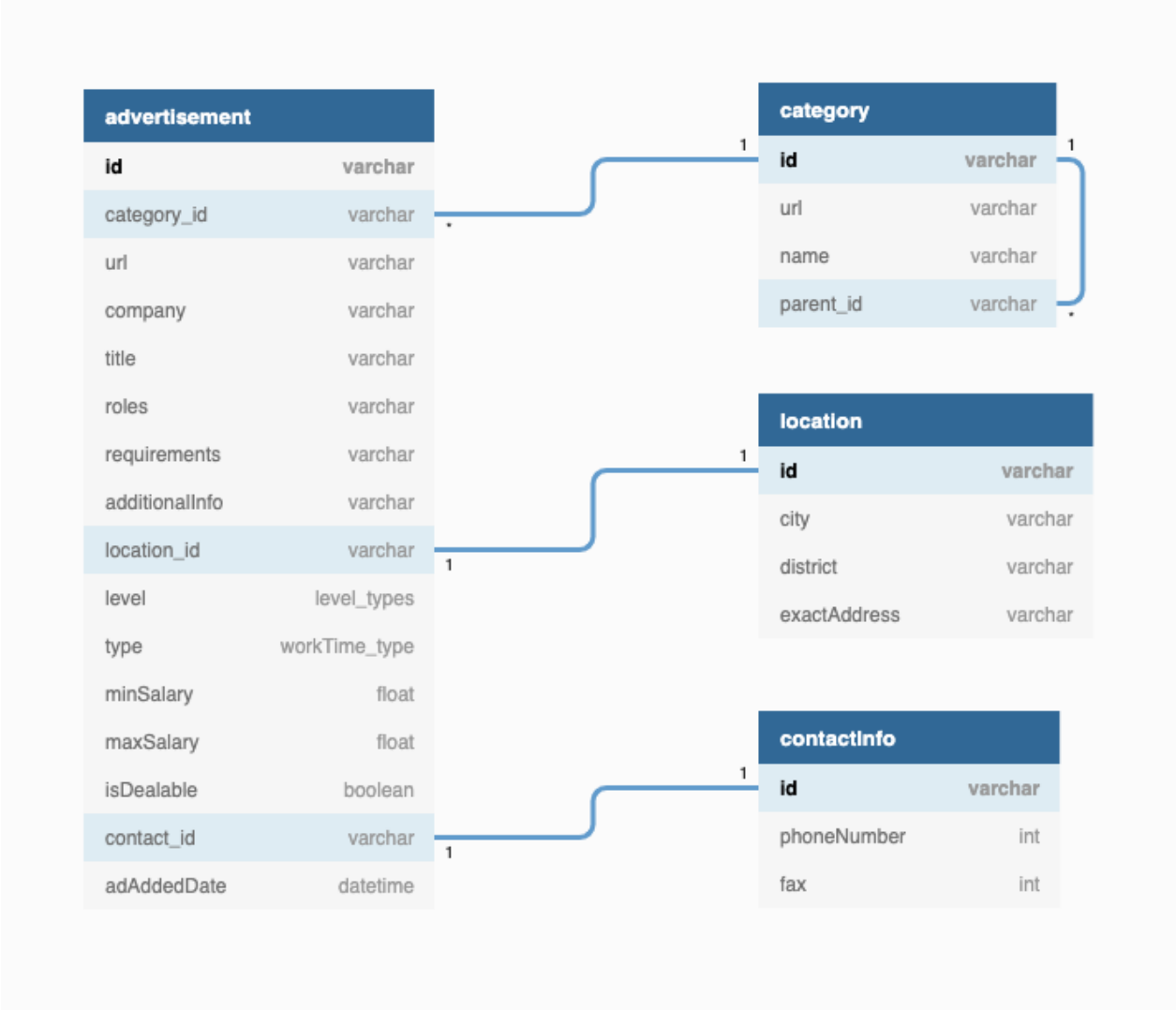
Чатбот системийн use-case диаграммыг байдлаар тодорхойлов [4].



Зураг 3.3: Use Case диаграмм

4. СИСТЕМИЙН ЗОХИОМЖ

4.1 Өгөгдлийн сангийн диаграмм



Зураг 4.1: Өгөгдлийн сангийн диаграмм

4.2 Өгөгдлийн элемент

Чатбот системийн өгөгдлийн сангийн диаграммд харуулсан хүснэгтүүдэд агуулагдах мэдээлэл болон үүргийн талаар дэлгэрэнгүй тайлбарласан болно.

4.2.1 advertisement - Ажлын байрны зар

Ажлын байрны зар нь ямар категори буюу ангилалд, ямар холбоо барих хаягийн хамтаар хадгалагдаж буй мэдээлэл болон бусад дэлгэрэнгүй мэдээллийг харуулсан байна.

Table 4.1: advertisement хүснэгт

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон утга	Тайлбар
1	**id**	PK	varchar	not null	Ажлын байрны зарын дахин давтагдашгүй дугаар
2	category_id	FK	varchar	not null	Ажлын байрны зард хамаарах ангиллын дугаар
3	url		varchar	not null	Ажлын байрны зарын хаяг
4	company		varchar	not null	Ажил олгогч компани / хүн
5	title		varchar	not null	Ажлын зарын гарчиг
6	roles		varchar	null	Гүйцэтгэхүндсэн үүрэг
7	requirements		varchar	null	Ажлын байранд тавигдах шаардлага
8	additionalInfo		varchar	null	Нэмэлт мэдээлэл
9	location_id	FK	varchar	not null	Ажлын байрны зард хамаарах ангиллын дугаар
10	level		level_types	null	Ажлын түвшин
11	type		workTime_type	null	Ажиллах цагийн төрөл

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон утга	Тайлбар
12	minSalary		float	null	Доод цалин
13	maxSalary		float	null	Дээд цалин
14	isDealable		boolean	null	Тохиролцох эсэх
15	contact_id	FK	varchar	not null	Ажлын байрны зард хамаарах холбоо барих хаягийн дугаар
16	adAddedDate		datetime	not null	Зар нийтэлсэн огноо

Энд *level* буюу ажлын түвшин, *type* буюу ажлын цагийн өгөгдлийн төрлийг тодорхойлохдоо дараах байдлаар зааж өгсөн.

Enum level_types буюу ажлын түвшний шаардлага нь дараах үндсэн 4 өгөгдлийн төрлөөс хамаарна:

- student - Оюутан / дадлагажигч
- professional - Мэргэжлийн
- occupasionDoesntRequire - Мэргэжил шаардахгүй
- intermediateManagemet - Дунд шатны удирдлага
- topLevelManagemet - Дээд шатны удирдлага

workTime_type буюу ажиллах цагийн нөхцөл нь дараах үндсэн 4 өгөгдлийн төрлөөс хамаарна:

- shift - Ээлжийн
- fullTime - Бүтэн цагийн
- halfTime - Хагас цагийн
- contract - Гэрээт / зөвлөх

- seasonal - Улирлаар

4.2.2 category - Ангилал

Ажлын байрны зарын бүх ангиллуудын хаяг болон нэрийн мэдээллийг хадгалах хүснэгт юм. Ангиллууд нь дэд ангилал байж болох учир түүнийг эцэг ангиллын дугаарыг хадгалах байдлаар зохиомжлов.

Table 4.2: category хүснэгт

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон Утга	Тайлбар
1	**id**	PK	varchar	not null	Ажлын байрны зарын ангиллын дугаар
2	url		varchar	not null	Ангиллын хаяг
3	name		varchar	not null	Ангиллын нэр
4	parent_id	FK	varchar	null	Эцэг ангиллын дугаар

4.2.3 location - Байршил

Ажлын байрны байршил болон хот, аймаг, дүүргийн дэлгэрэнгүй өгөгдлийг хадгална.

Table 4.3: location хүснэгт

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон Утга	Тайлбар
1	**id**	PK	varchar	not null	Ажлын байрны зарын хаягийн дугаар
2	city		varchar	null	Ажлын байрны зарын байрших хот, аймгийн нэр

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон Утга	Тайлбар
3	district		varchar	null	Ажлын байрны зарын байрших дүүрэг, сумын нэр
4	exactAddress		varchar	null	Дэлгэрэнгүй хаяг

4.2.4 contactInfo - Холбоо барих

Ажлын байр олгогчийн хаягийн дэлгэрэнгүй өгөгдлийг хадгалана.

Table 4.4: contactInfo хүснэгт

№	Баганын нэр	Түлхүүр өгөгдөл	Өгөгдлийн төрөл	Хоосон Утга	Тайлбар
1	**id**	PK	varchar	not null	Ажил олгогчтой холбоо барих хаягийн дугаар
2	phoneNumber		int	null	Ажил олгогчийн утасны дугаар
3	fax		int	null	Ажил олгогчийн факс дугаар

4.3 Өгөгдлийн сангийн холбоосын тайлбар

- Нэг ангилал буюу категорид олон ажлын байрны зар байж болно.
- Нэг ангилал буюу категорид олон категори байж болно.
- Нэг ажлын байрны зард нэг байршлын мэдээлэл байна.
- Нэг ажлын байрны зард нэг холбоо барих хаягийн мэдээлэл байна.

5. ХЭРЭГЖҮҮЛЭЛТ, ҮР ДҮН

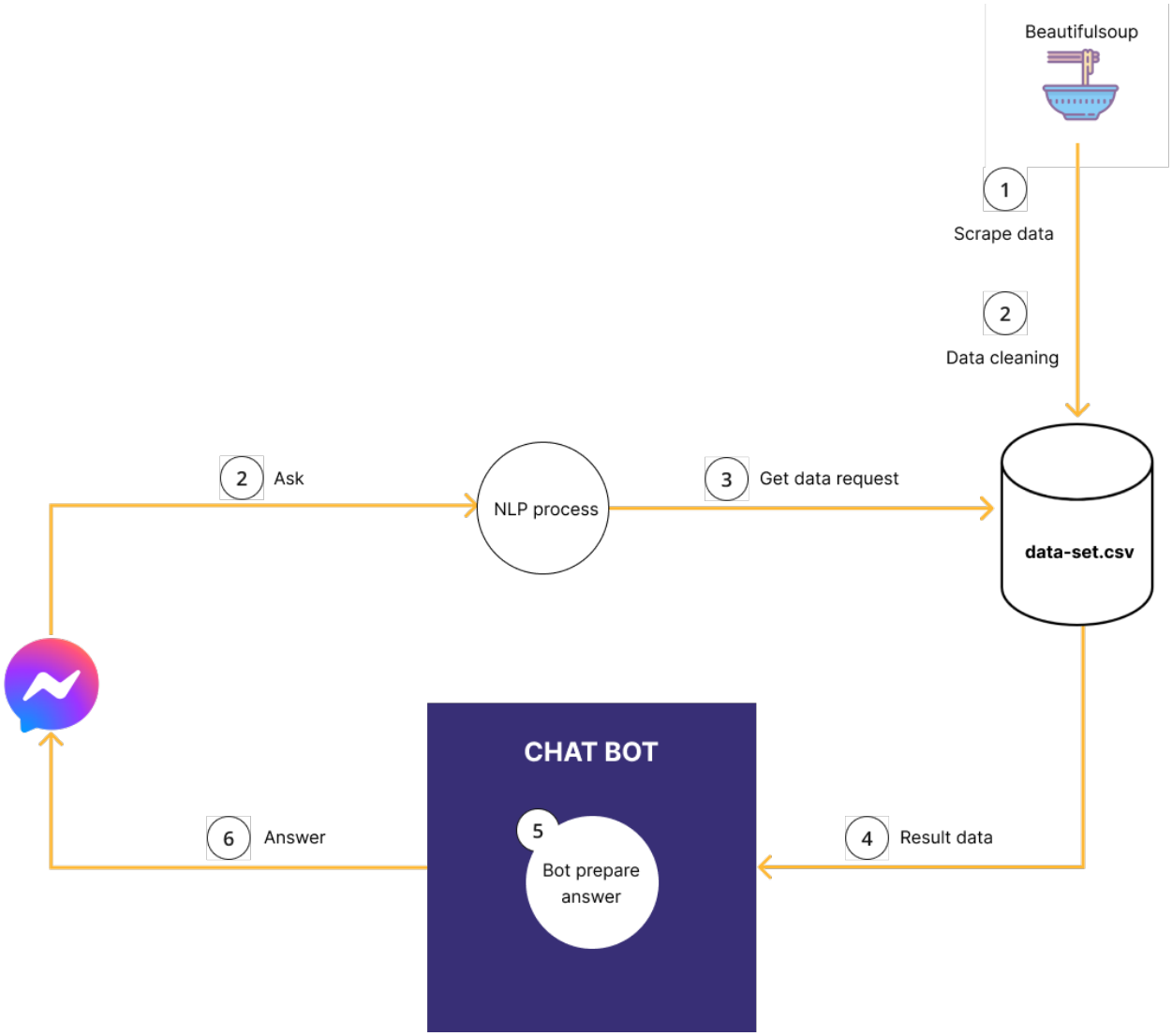
5.1 Хөгжүүлсэн байдал

Чатбот системийн хөгжүүлэлтийг хийхдээ шаардлагууд дээр үндэслэн, үечилсэн төлөвлөгөө болон шаардлагатай хөгжүүлэлтийг дэс дараалалтайгаар хийж гүйцэтгэсэн.

- Өгөгдөл цуглуулах
- Өгөгдлийг нэгтгэх, цэвэрлэх
- Системийн шаардлага, үйл ажиллагааг тодорхойлох
- Өгөгдөлд анализ хийх
- Эх хэлний боловсруулалт хийх
- Чатбот хөгжүүлэх

гэсэн дарааллын дагуу хөгжүүлэлтийг хийсэн болно.

Доорх зурагт чатбот системийн үндсэн процессийн зураглал харагдаж байна.



Зураг 5.1: Үндсэн процесс зураглал

5.1.1 Өгөгдөл цуглуулах

Үндсэн ашиглагдах өгөгдөл болох ажил олгогчид, ажлын байрны өгөгдлийг **zangia.mn**-ээс BeautifulSoup ашиглан авсан. Эхлээд вебсайтынхаа HTML бүтцийг нь судалж, авах өгөгдлийнхөө класс утгуудыг (className) олж авах нь зөв юм. Вебсайтаас өгөгдөл цуглуулах 2 үндсэн арга байдгаас өгөгдлийг олж илрүүлж, хаягийг цуглуулах (data crawling) аргаар бүх ангиллуудын хаяг (url)-уудын түүж авна. Харин data scraping нь тэр хооронд олсон бүх хаягуудаараа явж

хэрэгтэй агуулгыг цуглуулна. [1]

```
initialUrl = 'https://www.zangia.mn/'
today = str(date.today())
# all categories set
categorySet = set()
# all advertisement's link set
adUrlDict = {}
# all ads object set
adsSet = set()

# scrape initial links
soup = useScrape(initialUrl)
navigatorList = soup.find_all('div', class_='filter')
for navigator in navigatorList:
    if navigator.find('h3').text.strip() != 'Salbar, mergejil':
        continue
    # ALL CATEGORY LINKS
    categoryList = navigator.find_all('div')

for categoryItem in categoryList:
    categories = categoryItem.find('a')
    url = initialUrl + categories['href']
    tempCategory = Category(url, categories.text, '')
    soup = useScrape(url)
    subCategory = soup.find('div', class_='pros')
    # ALL SUBCATEGORY LINKS
    subCategoryList = subCategory.find_all('a')
```

[1] Кодын жишээг оруулахад utf-8 формат танихгүй байсан тул монголоос галиглаж бичсэн болно.

```
    for subCategoryItem in subCategoryList:
        subCategoryUrl = initialUrl + subCategoryItem['href']
        tempSubCategory = Category(
            subCategoryUrl, subCategoryItem.text, tempCategory.name)
        categorySet.add(tempSubCategory)
```

Код 5.1: Data Link crawling

Дээрх код нь эхлээд вебсайтруу орж “filter” класс доторх “Салбар, мэргэжил” гэсэн хэсгээс бүх эцэг категориудыг data crawling хийж авч байна. Үүний дараа хүүхэд категориудыг олж categorySet дотор бүх хаягуудыг хийж хадгалж байна. [2] Энд categorySet set-ийн элемент нь category төрлийн объект бөгөөд өгөгдлийн сангийн диаграм дээр тодорхойлж өгсөн байгаа. Ингэснээр data crawling-ийг зогсоож, цуглуулсан хаягаасаа өгөгдлөө цуглуулъя.

```
for categoryItem in categorySet:
    if categoryItem.parentId == '':
        continue
    soup = useScrape(categoryItem.url)
    hasPagination = soup.find('div', class_='page-link')
    pagesUrl = []
    if hasPagination != None:
        pagesUrl = createLinkList(hasPagination, categoryItem.url)
    else:
        pagesUrl.append(categoryItem.url)
    for pageUrl in pagesUrl:
        soup = useScrape(pageUrl)
        ads = soup.find_all('div', class_='ad')
        # CREATE UNIQUE AD DICTIONARY
        for ad in ads:
```

[2] Python хэлний set өгөгдлийн төрөл нь давхацахгүй утгуудын хүснэгт гэж хэлж болно.

```
            adUrl = initialUrl+ad.find('a', class_=None)['href']
            adUrlDict[adUrl] = categoryItem
    pagesUrl.clear()
```

Код 5.2: Өгөгдөл цуглуулах

Дээрх кодод бүх хүүхэд категориудын дотор агуулагдаж буй зарын мэдээллийг цуглуулж байна. Ингэхдээ эхлээд категори доторх өгөгдлүүд нь хуудаслагдсан (pagination) байх боломжтой бөгөөд хэрэв олон хуудастай байвал хаягуудыг нь угсарч тэдгээрээс ч мөн өгөгдлийг нь цуглуулах ёстой юм.

```
from .scrape import UseBeautifulSoup as useScrape

def createLinkList(pagination, url) -> array:
    linkList = []
    total = int(useRegex(pagination.find_all('a')[-1]['href']))

    for i in range(total + 1):
        if i == 0:
            continue
        link = url + '/pg.' + str(i)
        linkList.append(link)
    return linkList
```

Код 5.3: Хуудаслалтыг задлах

Энэ хэсэгт хуудаслан дугаарласан хэсгийн хамгийн сүүлийн тоог авч *createLinkList* функцруу дамжуулснаар тухайн категорийн бүх өгөгдлийг цуглуулах боломж үүсч байгаа юм. Ингээд дахин data crawling хийж бүх хаягуудыг цуглуулж энэ удаад dictionary үүсгэж зарын хаягуудыг хадгалсан. Энд dictionary үүсгэхдээ хаягийг нь түлхүүр (key) болгож категори объектыг нь утга(value) болгож хадгалсан. Мэдээж хэрэг dictionary нь түлхүүр давхцахаас сэргийлдэг тул

бид ямар нэгэн байдлаар нэг зарын өгөгдлийг 2 удаа цуглуулах эрсдэлгүй болж байна. [3]

Харин одоо үүсгэсэн dictionary-оо ашиглан өгөгдлөө CSV файлруугаа бичихэд ашиглаж болно.

```
for adUrl in adUrlDict:
    print(adUrl)
    try:
        tempAdItem = useAdScrape(adUrl)
        tempAdItem.setCategory(adUrlDict[adUrl])
        file.write(
            tempAdItem.category.parentId+'\t' +
            tempAdItem.category.name+'\t' +
            tempAdItem.url+'\t' +
            tempAdItem.company+'\t' +
            tempAdItem.title+'\t' +
            tempAdItem.roles+'\t' +
            tempAdItem.requirements+'\t' +
            tempAdItem.additionalInfo+'\t' +
            tempAdItem.city+'\t' +
            tempAdItem.district+'\t' +
            tempAdItem.level+'\t' +
            tempAdItem.type+'\t' +
            tempAdItem.minSalary+'\t' +
            tempAdItem.maxSalary+'\t' +
            tempAdItem.isDealable+'\t' +
            tempAdItem.address+'\t' +
            tempAdItem.phoneNumber+'\t' +
            tempAdItem.fax+'\t' +
```

[3]Нэг зарын өгөгдлийг цуглуулахад интернетийн хурдаас хамааран 0.2-оос 0.5 секунтын хугацаа зарцуулдаг

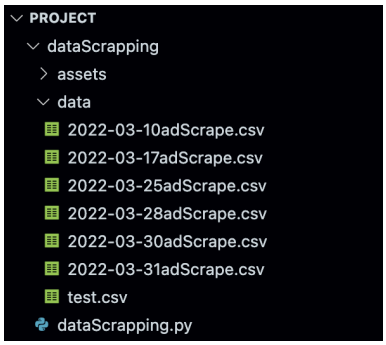
```
25              tempAdItem.adAddedDate+'\n')
26          del tempAdItem
27      except:
28          print('Ad writing error')
29  file.close()
```

Код 5.4: CSV файлруу хадгалах

Дээрх код нь энгийн python програм файлтай харьцаж өөрт цуглуулсан өгөгдлөө хадгалж байна. Нийт өгөгдлийн хүснэгтийг энд [4] оруулав.

5.1.2 Ажлын байрны өгөгдөл

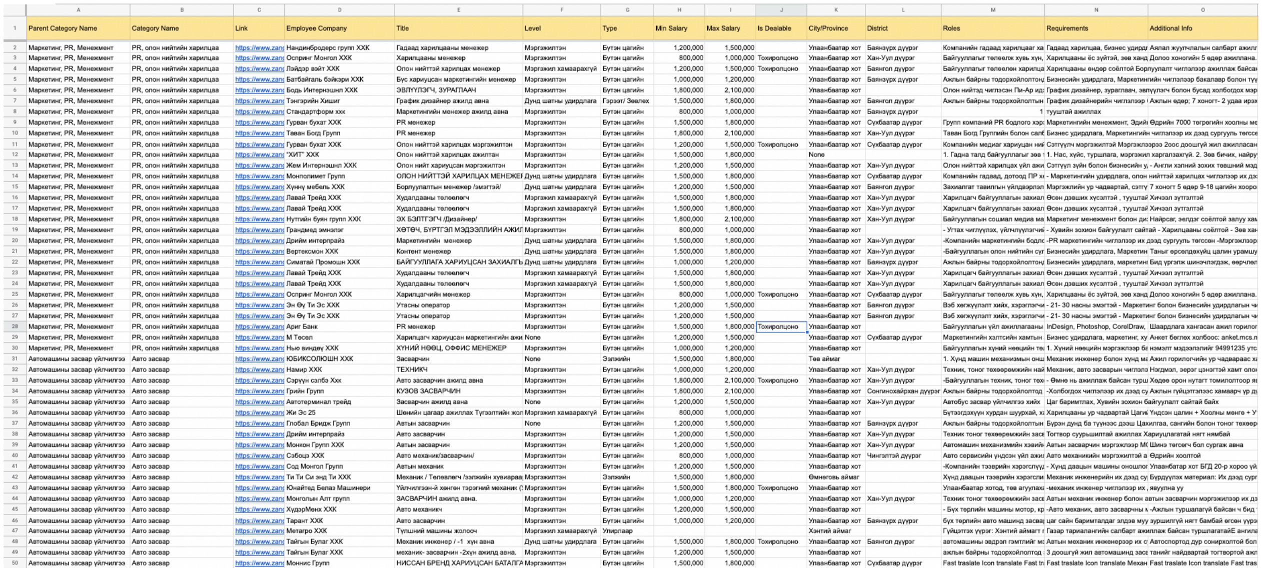
Zangia.mn ажлын байрны зарын вебсайтаас бакалаврын судалгааны ажлын үечилсэн төлөвлөгөөний дагуу 3 сарын 10-аас эхлэн өгөгдөл цуглуулсан билээ. Цуглуулж, өгөгдлийн сангийн зохиомжийн дагуу дараах утгуудын мэдээллийг CSV файлуудад хадгалж аваад байна. Доорх зурагт хамгийн



Зураг 5.2: Цуглуулсан өгөгдлийн файлууд

сүүлд буюу 3 сарын 31нд өгөгдлийн цуглуулга хийж 9174 өгөгдлийн excel хэлбэрт оруулсныг харж болж байна.

[4] https://docs.google.com/spreadsheets/d/1rtATUKhUlleIKaWgFGvqiUWMipsrv-aCWZk-tYmzezU/edit?usp=sharing



Зураг 5.3: Data set

5.1.3 *Өгөгдлийг нэгтгэх, цэвэрлэх*

Ихэнх цуглуулсан өгөгдөл нь өгөгдлийн сангийн диаграмын дагуу амжилттай цуглуулсан бөгөөд дүн шинжилгээ хийх боломжтой өгөгдлүүдийг тусад нь хадгалж ашигласан болно. Data scrape хийх явцад бүх өгөгдлийг *string* хэлбэрээр цуглуулсан бол өгөгдөлд дүн шинжилгээ хийх явцад энэ нь тохиромжгүй тул цалинг *float* тохиролцох эсэхийг *boolean* болгож өөрчлөв.

```
def cleanSalary(salary) -> float:
    if(isinstance(salary, str)):
        return float(salary.replace(',', ''))
    return None


def cleanDealable(deal) -> bool:
    if(deal == '         '):
        return True
    return False
```

```
11
12
13 def cleanLocation(location) -> str:
14     if isinstance(location, str) and location != 'None':
15         return location
```

Код 5.5: Өгөгдөл цэвэрлэх функц

	A	B	C	D	E	F	G	H	I	J	K
1		branch	employee	jobTitle	level	type	minSalary	maxSalary	isDealable	city	district
2	0	PR, олон нийтийн харилцаа	Нандинбродерс	Гадаад харилца	Мэргэжилтэн	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Баянзүрх дүүрэг
3	1	PR, олон нийтийн харилцаа	Оспринг Монго	Харилцааны ме	Мэргэжилтэн	Бүтэн цагийн	800000	1000000	TRUE	Улаанбаатар хот	Хан-Уул дүүрэг
4	2	PR, олон нийтийн харилцаа	Лэйдэр вэйт ХХ	Олон нийттэй х	Мэргэжил хама	Бүтэн цагийн	1200000	1500000	TRUE	Улаанбаатар хот	Баянгол дүүрэг
5	3	PR, олон нийтийн харилцаа	Батбайгаль бэй	Бүс хариуцсан	Мэргэжилтэн	Бүтэн цагийн	1000000	1200000	FALSE	Улаанбаатар хот	Баянзүрх дүүрэг
6	4	PR, олон нийтийн харилцаа	Бодь Интернэш	ЭВЛҮҮЛЭГЧ, ЗУ	Мэргэжилтэн	Бүтэн цагийн	1800000	2100000	FALSE	Улаанбаатар хот	
7	5	PR, олон нийтийн харилцаа	Тэнгэрийн Хиш	График дизайне	Дунд шатны уд	Гэрээт/ Зөвлөх	1500000	1800000	FALSE	Улаанбаатар хот	Баянгол дүүрэг
8	6	PR, олон нийтийн харилцаа	Стандартформ	Маркетингийн м	Мэргэжилтэн	Бүтэн цагийн	800000	1000000	FALSE	Улаанбаатар хот	Баянзүрх дүүрэг
9	7	PR, олон нийтийн харилцаа	Гурван бухат ХХ	PR менежер	Мэргэжилтэн	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Сүхбаатар дүүрэг
10	8	PR, олон нийтийн харилцаа	Таван Богд Груг	PR менежер	Мэргэжилтэн	Бүтэн цагийн	1800000	2100000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
11	9	PR, олон нийтийн харилцаа	Гурван бухат ХХ	Олон нийттэй х	Мэргэжилтэн	Бүтэн цагийн	1200000	1500000	TRUE	Улаанбаатар хот	Сүхбаатар дүүрэг
12	10	PR, олон нийтийн харилцаа	Жем Интернэш	Олон нийт хари	Мэргэжилтэн	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
13	11	PR, олон нийтийн харилцаа	Монполимет Гр	ОЛОН НИЙТТЭ	Дунд шатны уд	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Сүхбаатар дүүрэг
14	12	PR, олон нийтийн харилцаа	Хүннү мебель Х	Борлуулалтын м	Дунд шатны уд	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Баянгол дүүрэг
15	13	PR, олон нийтийн харилцаа	Лавай Трейд ХХ	Худалдааны төл	Мэргэжил хама	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
16	14	PR, олон нийтийн харилцаа	Лавай Трейд ХХ	Худалдааны төл	Мэргэжил хама	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
17	15	PR, олон нийтийн харилцаа	Нутгийн буян гр	ЭХ БЭЛТГЭГЧ /	Мэргэжилтэн	Бүтэн цагийн	1800000	2100000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
18	16	PR, олон нийтийн харилцаа	Грандмед эмнэ	ХӨТӨЧ, БУРТГ	Мэргэжилтэн	Бүтэн цагийн	800000	1000000	FALSE	Улаанбаатар хот	
19	17	PR, олон нийтийн харилцаа	Дрийм интерпр	Маркетингийн м	Дунд шатны уд	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
20	18	PR, олон нийтийн харилцаа	Вертексмон ХХ	Контент менеж	Дунд шатны уд	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
21	19	PR, олон нийтийн харилцаа	Симатай Промс	БАЙГУУЛЛАГА	Дунд шатны уд	Бүтэн цагийн	1000000	1200000	FALSE	Улаанбаатар хот	Баянзүрх дүүрэг
22	20	PR, олон нийтийн харилцаа	Лавай Трейд ХХ	Худалдааны төл	Мэргэжил хама	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
23	21	PR, олон нийтийн харилцаа	Лавай Трейд ХХ	Худалдааны төл	Мэргэжил хама	Бүтэн цагийн	1500000	1800000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
24	22	PR, олон нийтийн харилцаа	Оспринг Монго	Харилцагчийн м	Мэргэжилтэн	Бүтэн цагийн	800000	1000000	TRUE	Улаанбаатар хот	Сүхбаатар дүүрэг
25	23	PR, олон нийтийн харилцаа	Эн Өү Ти Эс ХХ	Утасны операто	Мэргэжилтэн	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Баянгол дүүрэг
26	24	PR, олон нийтийн харилцаа	Эн Өү Ти Эс ХХ	Утасны операто	Мэргэжилтэн	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Баянгол дүүрэг
27	25	PR, олон нийтийн харилцаа	Ариг Банк	PR менежер	Мэргэжилтэн	Бүтэн цагийн	1500000	1800000	TRUE	Улаанбаатар хот	
28	26	PR, олон нийтийн харилцаа	М Төсөл	Харилцагч хари	None	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Сүхбаатар дүүрэг
29	27	PR, олон нийтийн харилцаа	Нью виндөү ХХ	ХҮНИЙ НӨӨЦ,	Мэргэжилтэн	Бүтэн цагийн	1000000	1200000	FALSE	Улаанбаатар хот	
30	28	Авто засвар	ЮБИКСОЛЮШ	Засварчин	None	Ээлжийн	1500000	1800000	FALSE	Төв аймаг	
31	29	Авто засвар	Намир ХХК	ТЕХНИКЧ	Мэргэжилтэн	Бүтэн цагийн	1000000	1200000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг
32	30	Авто засвар	Сэрүүн сэлбэ Х	Авто засварчин	Мэргэжилтэн	Бүтэн цагийн	1800000	2100000	TRUE	Улаанбаатар хот	Хан-Уул дүүрэг
33	31	Авто засвар	Грийн Групп	КУЗОВ ЗАСВАР	Мэргэжилтэн	Бүтэн цагийн	1800000	2100000	FALSE	Улаанбаатар хот	Сонгинохайрхан дүүрэг
34	32	Авто засвар	Автотерминал т	Засварчин ажил	None	Бүтэн цагийн	1200000	1500000	FALSE	Улаанбаатар хот	Хан-Уул дүүрэг

Зураг 5.4: Өгөгдлийг цэвэрлэж, өөрчлөлт хийсэн өгөгдөл

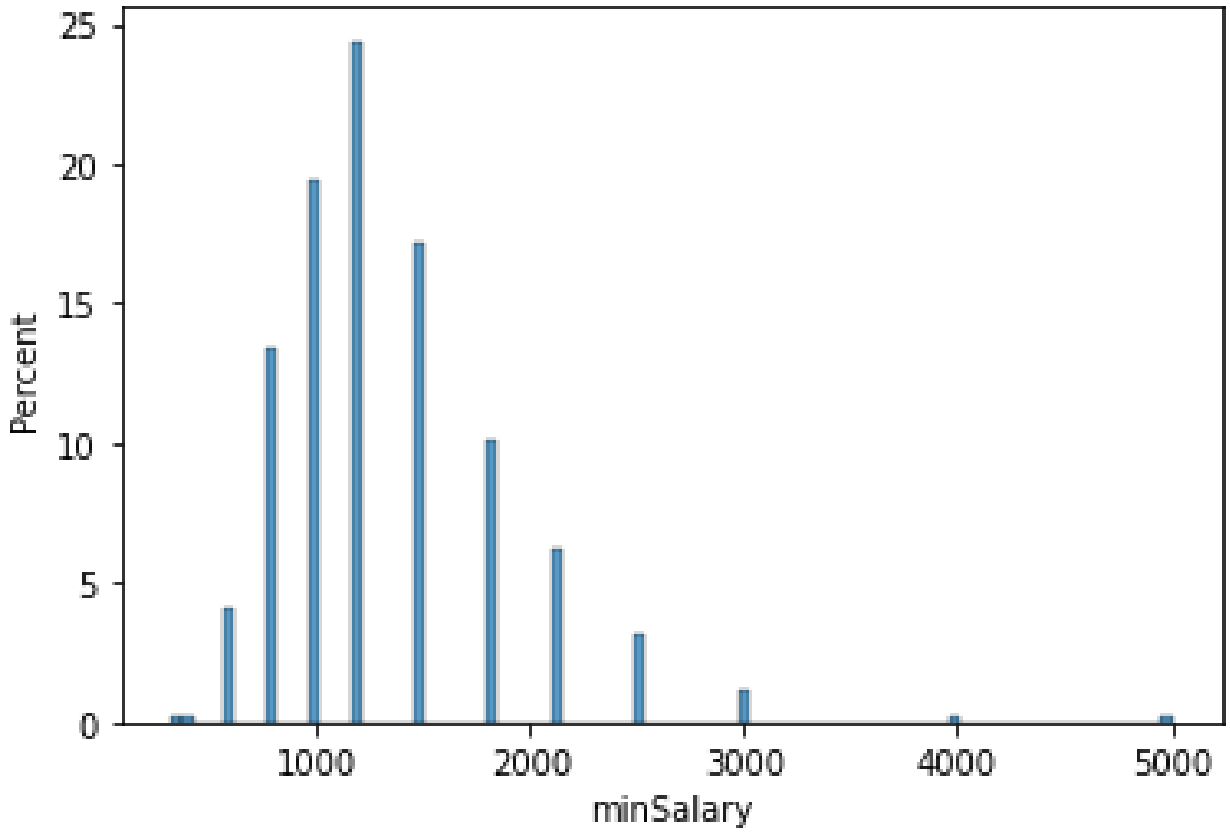
5.1.4 Өгөгдлийн статистик

Өөрчлөлт хийсэн өгөгдөл буюу нийт 8202 зарын өгөгдөл дээрээ дүн шинжилгээ хийж, цаашид үүнийгээ чатбот хөгжүүлэлт, асуулт хариултын загварт тусгахыг зорив.

Статистик - 1

Энэхүү графикт нийт цалингийн давхардсан утгууд нийт өгөгдлийн хэдэн хувийг эзэлж байгааг харуулж байна. Үүнээс харвал дундаж цалин 1 сая төгрөгөөс 1.5 сая төгрөгийн хооронд

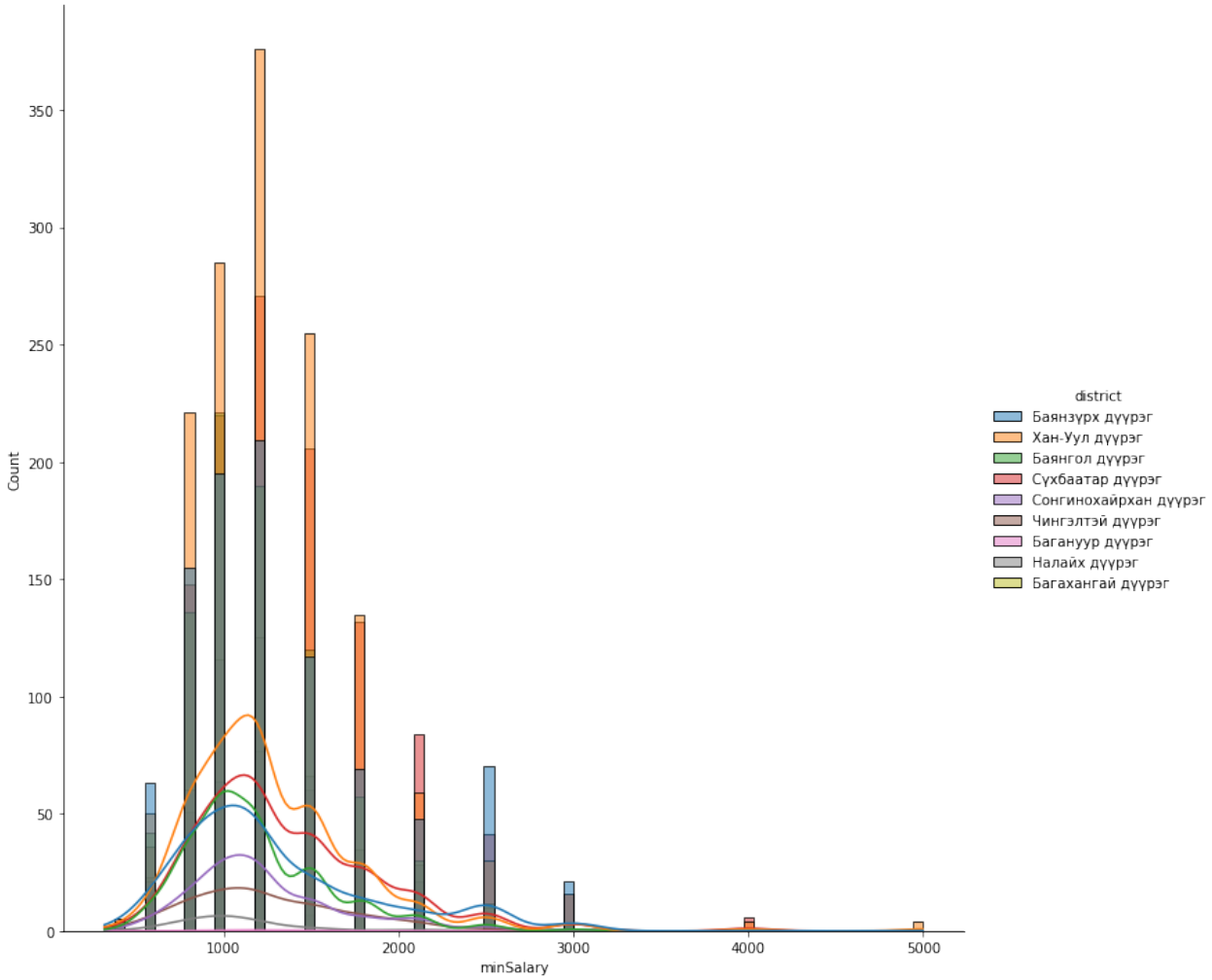
байгааг харж болж байна. Энэ нь нийт цалингийн хувийн 50-аас 55-ийг эзэлж байна.



Зураг 5.5: Өгөгдлийн статистик-1

Статистик - 2

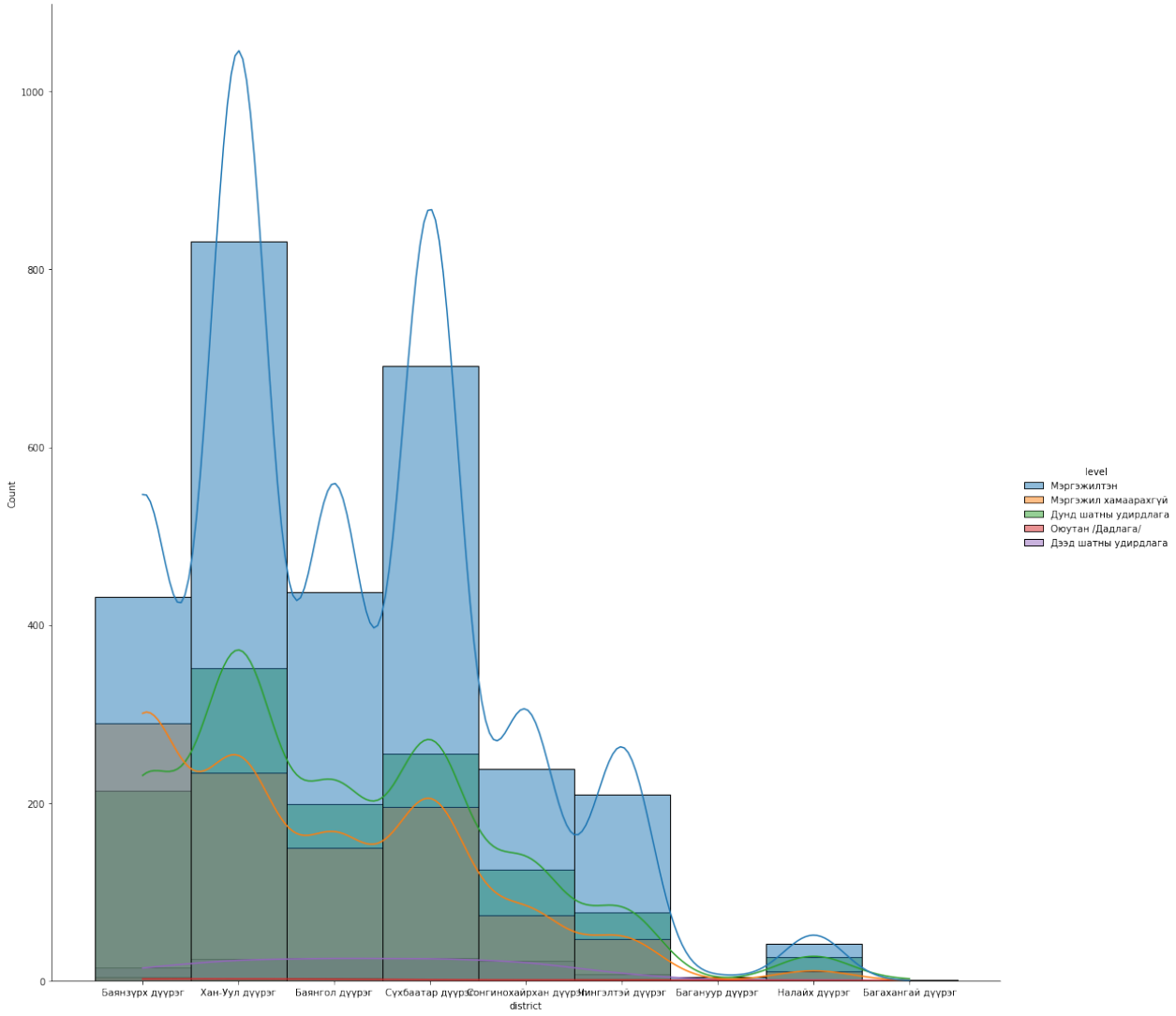
Доорх графикт нийт цалингийн давхардсан утгуудыг тоолж аль дүүрэгт хамгийн их байгааг өнгөөр нь дүрслэн харуулсан байна. Энэхүү графикаас харвал дундаж цалин буюу 1 сая төгрөгөөс 1.5 сая төгрөгөөр цалинжуулах олон ажлын байр санал болгож байгаа дүүрэг нь Хан-Уул болон Сүхбаатар дүүрэг байна. Ажлын байрны төвлөрөл болон их хотын бүтээгдэхүүн үйлдвэрлэлийн цэгийг Хан-Уул, Сүхбаатар дүүрэг гэж дүгнэж болохоор байна.



Зураг 5.6: Өгөгдлийн статистик-2

Статистик - 3

Өмнөх графикийн дүгнэлтийн адилаар хамгийн их ажлын санал болгож буй дүүрэг нь Хан-Уул, Сүхбаатар дүүрэг байх бөгөөд ажлын байрны шаардах түвшинг хамтад нь харуулсан байна. Үүнээс үзвэл, мэргэжилтэн болон дунд шатны удирдлагын орон тоо эрэлттэй байна гэж үзэж болно. Харин дадлагажигч болон дээд шатны удирдлагын эрэлт харьцангуй бага байгааг харж болж байна.



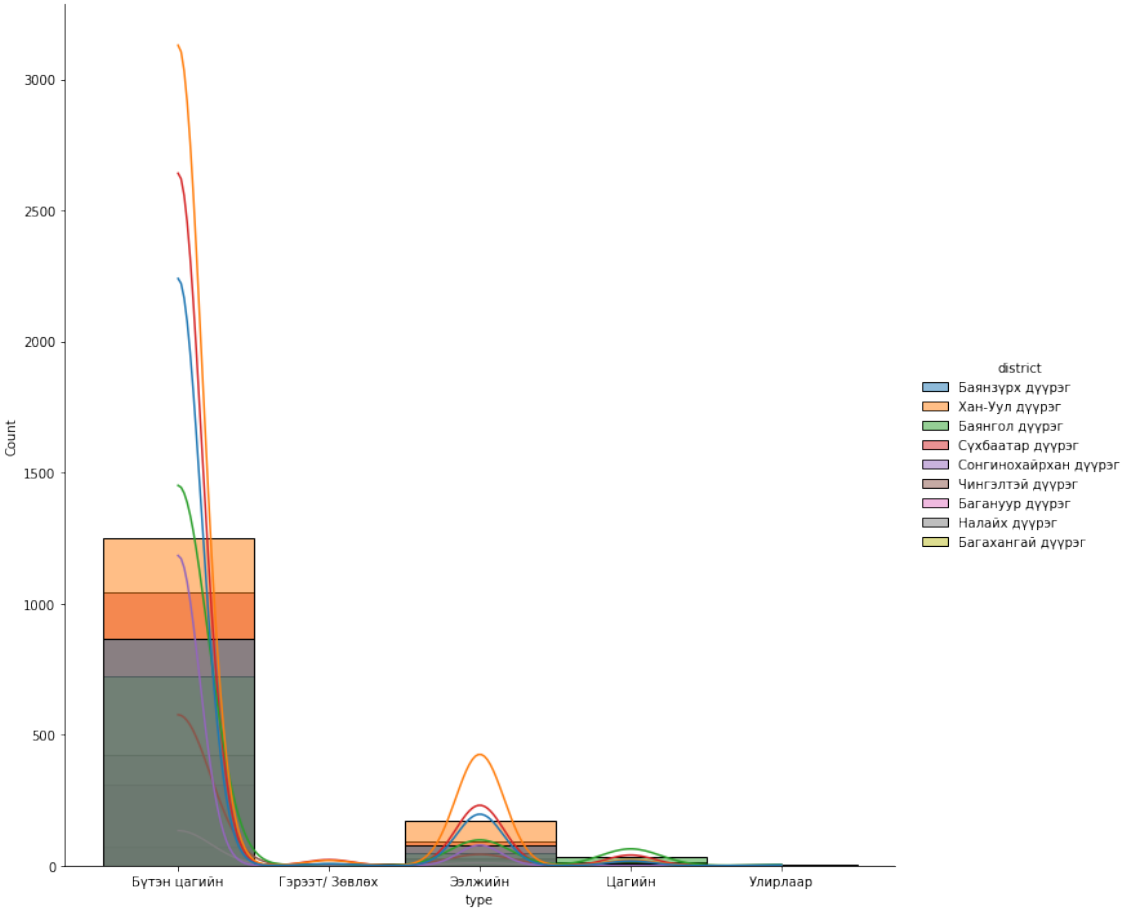
Зураг 5.7: Өгөгдлийн статистик-3

Статистик - 4

Энэхүү графикт дүүргүүд харгалзан ямар төрлийн цагийн хуваарьтай ажил санал болгож байгаа болон тэдгээрийн тоотой нь харьцуулан дүрсэлжээ. Эндээс бүтэн цагийн ажилтан болон ээлжийн төрлийн ажлын байр ихэнх хувийг эзэлж байгааг харлаа.

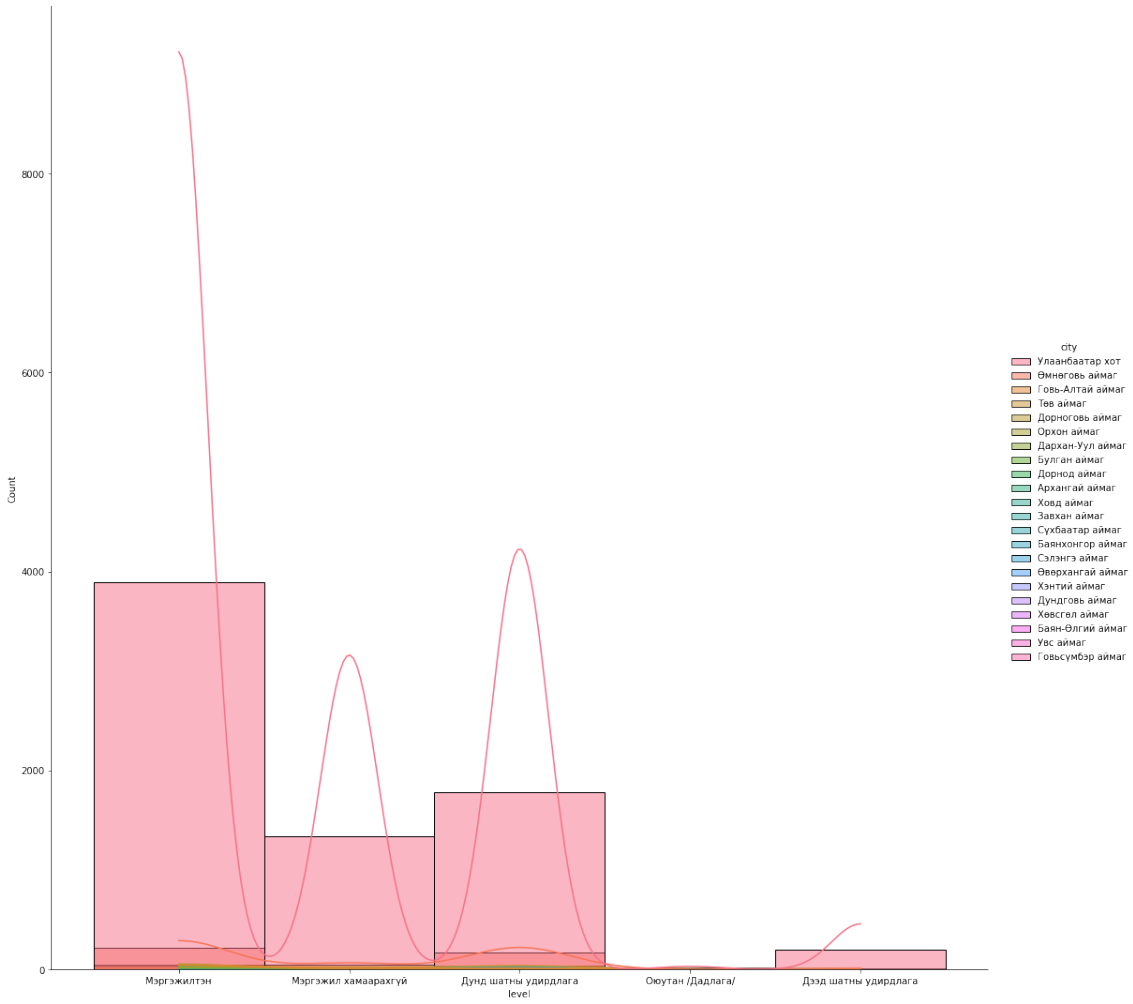
Статистик - 5

Доорх графикаас ажил олгогчид ямар төрлийн цагийн хуваарьтай, хаана ажил санал болгож байгааг 21 аймгаар бүсчлэн өнгөөр илэрхийлсэн байна. Үүнээс дүгнэвэл, Улаанбаатар хотод



Зураг 5.8: Өгөгдлийн статистик-4

нягтаршил маш өндөр байгаа бөгөөд ажлын байрны эрэлт аймгуудтай харьцуулахад маш өндөр байна.



Зураг 5.9: Өгөгдлийн статистик-5

Дүгнэлт

Бакалаврын судалгааны ажлаар “Ажил олгогчдын өгөгдлийн анализ систем дээр суурилсан чатбот” сэдвийн дагуу хөгжүүлэлтийг эхлүүлсэн бөгөөд уг судалгааны ажилд холбогдох онолын судалгаа, ашиглаж буй технологи, түүнийг илүү онолын мэдлэг болон системийн хөгжүүлэлтийн хэсгээс дэлгэрэнгүй тайлбарласан болно.

Bibliography

[1] Чатбот системийн тухай
https://www.engati.com/blog/types-of-chatbots-and-their-applications

[2] Өгүүлбэр хувиргалтын арга зүй
https://www.sbert.net/docs/quickstart.html

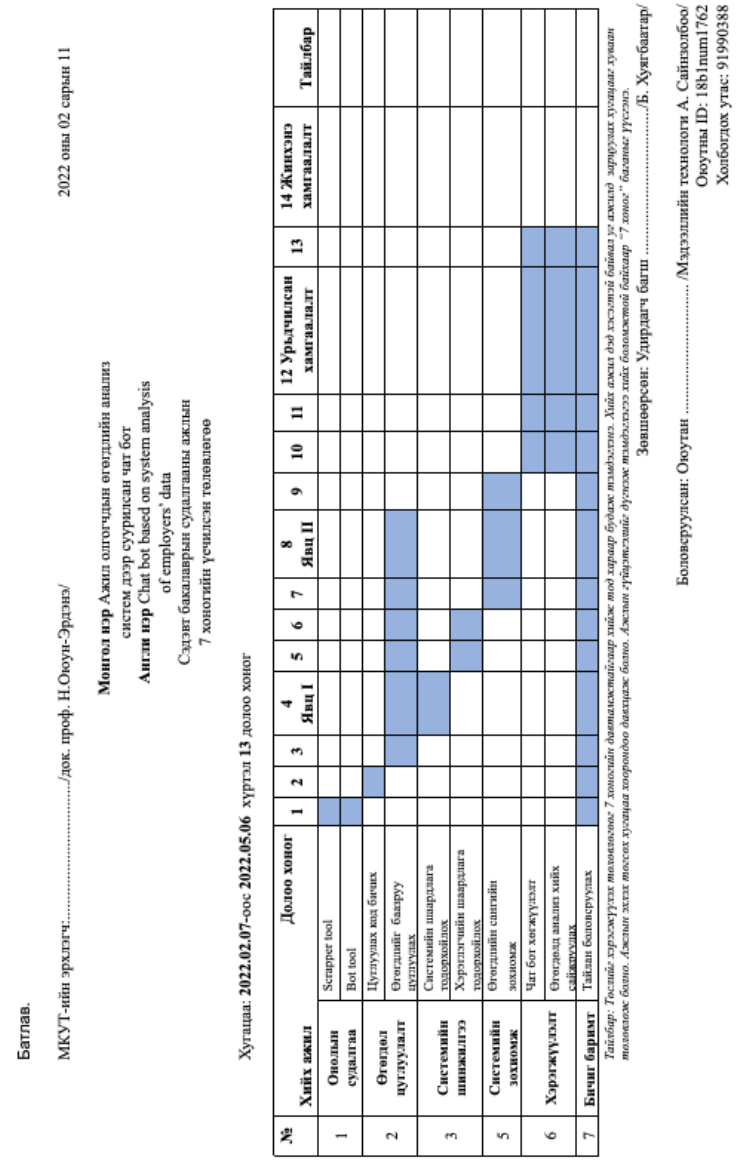
[3] Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks
https://arxiv.org/abs/1908.10084

[4] Use case diagram
https://app.diagrams.net/#G1jhom3sc_holt-X9XLALtQja_Gl_Eykhj

[5] Business Process Model Notation 2.0 диаграмм
https://cawemo.com/diagrams/ea037ec0-c1c5-4ab6-8262-521657472803--bpmn-2-0?v=960,418,1

[6] Өгөгдлийн сангийн диаграмм
https://dbdiagram.io/d/6249fb7cd043196e39e87451

А. ҮЕЧИЛСЭН ТӨЛӨВЛӨГӨӨ



Батлав.

МКУТ-ийн эрхлэгч:...../док. проф. Н.Оюун-Эрдэнэ/ 2022 оны 02 сарын 11

Монгол нэр Ажил олгогчын өгөгдлийн анализ
систем дээр суурилсан чат бот
Англи нэр Chat bot based on system analysis
of employers' data
Сэдэт бакалаврын судалгааны ажлын
7 хоногийн үечилсэн төлөвлөгөө

Хугацаа: **2022.02.07**-оос **2022.05.06** хүртэл **13** долоо хоног

№	Хийх ажил	Долоо хоног	1	2	3	4 Яац I	5	6	7	8 Яац II	9	10	11	12 Урьдчилсан хамгаалалт	13	14 Жинхэнэ хамгаалалт	Тайлбар
1	**Онолын судалгаа**	Scrapper tool															
		Bot tool															
2	**Өгөгдөл цуглуулалт**	Цуглуулах код бичих															
		Өгөгдлийг баяжуу цуглуулах															
3	**Системийн шинжилгээ**	Системийн шаардлага тодорхойлох															
		Хэрэглэгчийн шаардлага тодорхойлох															
5	**Системийн зохиомж**	Өгөгдлийн сангийн зохиомж															
		Чат бот хөгжүүлэлт															
6	**Хэрэгжүүлэлт**	Өгөгдөл анализ хийх сайжруулах															
7	**Бичиг баримт**	Тайлан боловсруулах															

Тайлбар: Тестийг хэрэгжүүлэх төлөвлөгөө 7 хоногийн давтамжтайгаар тийм тоо харгар будаж тэмдэглэнэ. Хийх ажил дуусгахаар байвал ус ажилд зарцуулах хугацаг хувиар төлөвлөж болно. Ажлын эхэл төгсөх хугацаа хоорондоо давхцаж болно. Ажлын гүйцэтгэлийг дүгнэж тэмдэглэсэн хийх боломжтой байхаар "7 хоног" баганас үүсгэнэ.

Зөвшөөрсөн: Удирдагч багш ...../Б. Хүнгбаатар/

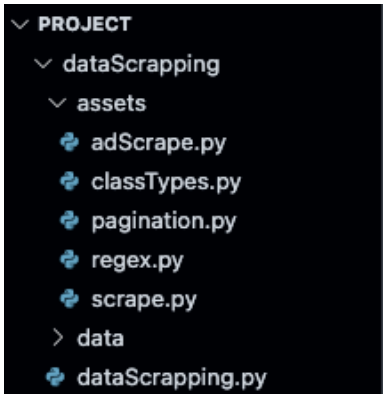
Боловсруулсан: Оюутан ...../Мэдээллийн технологи А. Сайнзолбоо/
Оюутны ID: 18b1num1762
Холбогдох утас: 91990388

Зураг А.1: Бакалаврын судалгааны ажлын үечилсэн төлөвлөгөө

B. КОДЫН ХЭРЭГЖҮҮЛЭЛТ

B.1 Өгөгдөл цуглуулалт

Өгөгдөл цуглуулах програм нь дараах бүтэцтэй байх бөгөөд assets доторх кодууд нь үндсэн кодыг ажлуулахад туслах функцууд байна.



Зураг B.1: Фолдерийн бүтэц

B.1.1 Үндсэн өгөгдлийг цуглуулах эх код

```
from datetime import date
import time
from assets.classTypes import Category
from assets.scrape import UseBeautifulSoup as useScrape
from assets.adScrape import advertisementScrape as useAdScrape
from assets.pagination import createLinkList as createLinkList

start_time = time.time()
initialUrl = 'https://www.zangia.mn/'
today = str(date.today())
# all categories set
categorySet = set()
# all advertisement's link set
adUrlDict = {}
# all ads object set
adsSet = set()

# scrape initial links
soup = useScrape(initialUrl)
navigatorList = soup.find_all('div', class_='filter')
for navigator in navigatorList:
    if navigator.find('h3').text.strip() != 'Salbar, mergejil':
        continue
    # ALL CATEGORY LINKS
    categoryList = navigator.find_all('div')
```

```
26
27  for categoryItem in categoryList:
28      categories = categoryItem.find('a')
29      url = initialUrl + categories['href']
30      tempCategory = Category(url, categories.text, '')
31      soup = useScrape(url)
32      subCategory = soup.find('div', class_='pros')
33      # ALL SUBCATEGORY LINKS
34      subCategoryList = subCategory.find_all('a')
35      for subCategoryItem in subCategoryList:
36          subCategoryUrl = initialUrl + subCategoryItem['href']
37          tempSubCategory = Category(
38              subCategoryUrl, subCategoryItem.text, tempCategory.name)
39          categorySet.add(tempSubCategory)
40
41  for categoryItem in categorySet:
42      if categoryItem.parentId == '':
43          continue
44      soup = useScrape(categoryItem.url)
45      hasPagination = soup.find('div', class_='page-link')
46      pagesUrl = []
47      if hasPagination != None:
48          pagesUrl = createLinkList(hasPagination, categoryItem.url)
49      else:
50          pagesUrl.append(categoryItem.url)
51      for pageUrl in pagesUrl:
52          soup = useScrape(pageUrl)
53          ads = soup.find_all('div', class_='ad')
54          # CREATE UNIQUE AD DICTIONARY
55          for ad in ads:
56              adUrl = initialUrl+ad.find('a', class_=None)['href']
57              adUrlDict[adUrl] = categoryItem
58      pagesUrl.clear()
59
60  file = open(today+'adScrape.csv', 'w', encoding='utf-8')
61  file.write('Parent Category Name' + '\t' +
62             'Category Name ' + '\t' +
63             'Link' + '\t' +
64             'Employee Company' + '\t' +
65             'Title' + '\t' +
66             'Roles' + '\t' +
67             'Requirements' + '\t' +
68             'Additional Info' + '\t' +
69             'City/Province' + '\t' +
70             'District' + '\t'
71             'Level' + '\t' +
72             'Type' + '\t' +
73             'Min Salary' + '\t' +
74             'Max Salary' + '\t' +
75             'Is Dealable' + '\t' +
76             'Address' + '\t' +
77             'Phone' + '\t' +
```

```
            'Fax' + '\t' +
            'Ad Added Date' + '\n')

for adUrl in adUrlDict:
    print(adUrl)
    try:
        tempAdItem = useAdScrape(adUrl)
        tempAdItem.setCategory(adUrlDict[adUrl])
        file.write(
            tempAdItem.category.parentId+'\t' +
            tempAdItem.category.name+'\t' +
            tempAdItem.url+'\t' +
            tempAdItem.company+'\t' +
            tempAdItem.title+'\t' +
            tempAdItem.roles+'\t' +
            tempAdItem.requirements+'\t' +
            tempAdItem.additionalInfo+'\t' +
            tempAdItem.city+'\t' +
            tempAdItem.district+'\t' +
            tempAdItem.level+'\t' +
            tempAdItem.type+'\t' +
            tempAdItem.minSalary+'\t' +
            tempAdItem.maxSalary+'\t' +
            tempAdItem.isDealable+'\t' +
            tempAdItem.address+'\t' +
            tempAdItem.phoneNumber+'\t' +
            tempAdItem.fax+'\t' +
            tempAdItem.adAddedDate+'\n')
        del tempAdItem
    except:
        print('Ad writing error')
file.close()
print("--- %s seconds ---" % (time.time() - start_time))
```

Код В.1: Бүх өгөгдлийг цуглуулах - dataScraping.py

В.1.2 Нэг зарын шаардлагатай бүх мэдээллийг цуглуулах код

```
import re
from .classTypes import Advertisement
from .scrape import UseBeautifulSoup as useScrape


def listScraper(sections, key) -> str:
    content = []
    for section in sections:
        subTitle = section.find('h2', class_=None).text
        if key != subTitle:
            continue
        div = section.find('div', class_=None)
        children = div.next_element

```

```
        while(children != None):
            try:
                content.append(textStrip(children.text))
                children = children.next_sibling
                continue
            except:
                print('An error occured')
            children = children.next_sibling
        content = [s for s in filter(listFunc, content)]
    if not content:
        return ''
    return ' '.join(content)


def textStrip(text) -> str:
    pattern = re.compile('[\r\n\xa0\t ]+', re.MULTILINE | re.IGNORECASE
        )
    return pattern.sub(' ', text.strip())


def listFunc(e):
    return len(e) != 0


def singleItemScraper(sections, key, subKey) -> str:
    for section in sections:
        subTitle = section.find('h2', class_=None).text
        if key != subTitle:
            continue
        div = section.find_all('div', class_=None)
        for item in div:
            if item.next_element.text == subKey:
                return textStrip(item.find('span').text)
    return 'None'


def salaryScraper(salary):
    isDealable = ''
    k = re.split(r'[^\d,]+', salary, 2, re.IGNORECASE)
    if len(k) < 2:
        [a] = k[0:1]
        return a, a
    [a, b] = k[0:2]
    if len(k) > 2:
        isDealable = '     '
    return a, b, isDealable


def locationScrapper(location):
    city = ''
    district = ''
    k = location.split(',')
```

```
    if len(k) < 2:
        city = k[0]
        return city, district
    [city, district] = k[0:2]
    return city, district


def advertisementScrape(url) -> Advertisement:
    soup = useScrape(url)
    advertisement = Advertisement(url, soup.find('h3').text.strip())
    companyTitle = soup.find('div', class_='nlp').find('td')
    for item in companyTitle:
        try:
            if item.name == None:
                advertisement.company = textStrip(item.text)
        except:
            print('Company name scrape error')
    # advertisement.company = textStrip(company)

    # all items
    sections = soup.find_all('div', class_='section')
    advertisement.roles = listScraper(
        sections, 'Guitsetgeh undsen uurerg')
    advertisement.requirements = listScraper(
        sections, 'Ajliin bairnii shaardlaga')
    advertisement.additionalInfo = listScraper(
        sections, 'Nemelt medeelel')
    advertisement.level = singleItemScraper(sections, 'Busad', 'Tuvshin
        ')
    advertisement.type = singleItemScraper(sections, 'Busad', 'Turul')
    minSalary, maxSalary, isDealable = salaryScraper(
        singleItemScraper(sections, 'Busad', 'Tsalin'))
    city, district = locationScrapper(
        singleItemScraper(sections, 'Busad', 'Bairshil'))
    advertisement.minSalary = minSalary
    advertisement.maxSalary = maxSalary
    advertisement.isDealable = isDealable
    advertisement.city = city
    advertisement.district = district
    advertisement.address = singleItemScraper(sections, '      ', '  ')
    advertisement.phoneNumber = singleItemScraper(
        sections, 'Holboo barih', 'Utas')
    advertisement.fax = singleItemScraper(
        sections, 'Holboo barih', 'Fax')
    advertisement.adAddedDate = singleItemScraper(
        sections, 'Zariin hugatsaa', 'Zar niitelsen ognoo')
    print(advertisement.additionalInfo)
    print('SINGLE AD SCRAPPING DONE!!!', url)

    return advertisement
```

Код B.2: Нэг зарын өгөгдлийг цуглуулах - adScrape.py

B.1.3 Цуглуулах өгөгдлийн төрөл

```
class Category:
    url = ''
    name = ''
    parentId = ''

    def __init__(self, url, name, parentId='None') -> None:
        self.url = url
        self.name = name
        self.parentId = parentId

    def getUrl(self) -> str:
        return self.url

class Advertisement:
    category = Category
    url = ''
    company = ''
    title = ''
    # ListInfo
    roles = ''
    requirements = ''
    additionalInfo = ''
    # OtherInfo
    city = ''
    district = ''
    level = ''
    type = ''
    minSalary = ''
    maxSalary = ''
    isDealable = ''
    # ContactInfo
    address = ''
    phoneNumber = ''
    fax = ''
    adAddedDate = ''

    def __init__(self, url, title) -> None:
        self.url = url
        self.title = title

    def setCategory(self, category) -> None:
        self.category = category
```

Код B.3: Өгөгдлийн төрөл - classTypes.py

B.1.4 BeautifulSoup scraper

```
from bs4 import BeautifulSoup
import requests
from urllib.error import HTTPError
```

```
4
5
6  def UseBeautifulSoup(url):
7      try:
8          response = requests.get(url)
9          response.raise_for_status()
10     except HTTPError as error:
11         print(error)
12     soup = BeautifulSoup(response.text, 'html.parser')
13     return soup
```

Код B.4: Scrape хийх функц - scrape.py

B.2 Өгөгдөл нэгтгэх, цэвэрлэх

```
1  from random import random
2  import requests
3  import pandas as pd
4  import numpy as np
5  import csv
6  import bs4 as BeautifulSoup
7  import random
8
9  with open('/Users/zolboo/Desktop/Bachelor/employementAnalysis/project/
       latestData.tsv', 'r', encoding='utf-8') as file:
10     app_lines = file.read().split('\n')
11
12 df = pd.read_csv(
13     '/Users/zolboo/Desktop/Bachelor/employementAnalysis/project/
           latestData.tsv', sep='\t')
14
15
16 def cleanSalary(salary) -> float:
17     if(isinstance(salary, str)):
18         return float(salary.replace(',', ''))
19     return None
20
21
22 def cleanDealable(deal) -> bool:
23     if(deal == '      '):
24         return True
25     return False
26
27
28 def cleanLocation(location) -> str:
29     if isinstance(location, str) and location != 'None':
30         return location
31     return None
32
33
34 def normalizeDataSet(data_set):
35     ret = pd.DataFrame(columns=['branch', 'employee', 'jobTitle', '
```

```
        level',
                        'type', 'minSalary', 'maxSalary', 'isDealable',
                           'city', 'district'])
    for index, row in data_set.iterrows():
        minSalary = cleanSalary(row['Min Salary'])
        maxSalary = cleanSalary(row['Max Salary'])
        location = cleanLocation(row['City/Provice'])
        print(type(row['District']), row['District'])
        if minSalary is None and maxSalary is None and location is None
           :
            continue
        dealable = cleanDealable(row['Is Dealable'])
        ret = ret.append({'branch': row['Category Name'],
                          'employee': row['Employee Company'],
                          'jobTitle': row['Title'],
                          'level': row['Level'],
                          'level': row['Type'],
                          'minSalary': minSalary,
                          'maxSalary': maxSalary,
                          'isDealable': dealable,
                          'city': location,
                          'district': row['District']
                          }, ignore_index=True)
    return ret


data_set = normalizeDataSet(df)
data_set.to_csv('data.csv')
```

Код В.5: Өгөгдөл цэвэрлэх - dataClean.py