

МОНГОЛ УЛСЫН ИХ СУРГУУЛЬ
МЭДЭЭЛЛИЙН ТЕХНОЛОГИ, ЭЛЕКТРОНИКИЙН СУРГУУЛЬ
МЭДЭЭЛЭЛ, КОМПЬЮТЕРИЙН УХААНЫ ТЭНХИМ

Гантөмөрийн Цогбадрах

Хэл солилцох аппликейшн
(Language exchange application)

Програм Хангамж (D 061302)
Бакалаврын судалгааны ажил

Улаанбаатар

2023 он

МОНГОЛ УЛСЫН ИХ СУРГУУЛЬ
МЭДЭЭЛЛИЙН ТЕХНОЛОГИ, ЭЛЕКТРОНИКИЙН СУРГУУЛЬ
МЭДЭЭЛЭЛ, КОМПЬЮТЕРИЙН УХААНЫ ТЭНХИМ

Хэл солилцох аппликейшн
(Language exchange application)

Програм Хангамж (D 061302)
Бакалаврын судалгааны ажил

Удирдагч: _____ Доктор Г.Ууганбаяр
Гүйцэтгэсэн: _____ Г. Цогбадрах (20B1NUM0477)

Улаанбаатар

2023 он

Зохиогчийн баталгаа

Миний бие Гантөмөрийн Цогбадрах "Хэл солилцох аппликейшн" 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	iv
ХҮСНЭГТИЙН ЖАГСААЛТ	v
КОДЫН ЖАГСААЛТ	vi
УДИРТГАЛ	1
Зорилго	1
Зорилт	1
Сэдэв сонгох үндэслэл	1
Ач холбогдол	2
БҮЛГҮҮД	4
1. СЭДВИЙН ЕРӨНХИЙ СУДАЛГАА	4
1.1 Ижил төсөөтэй систем	4
1.2 Ашиглах технологи	6
2. СИСТЕМИЙН ШААРДЛАГА	12
2.1 Шаардлагын шинжилгээ	12
2.2 UX/UI шаардлага	13
3. СИСТЕМИЙН АРХИТЕКТУР, ЗОХИОМЖ	15
3.1 Системийн архитектур	15
3.2 Ажлын явцын диаграм	17
3.3 UX/UI дизайн	19
3.4 Өгөгдлийн сан	27
4. ХЭРЭГЖҮҮЛЭЛТ	31
4.1 Хөгжүүлэлтийн орчныг бэлдэх	31
4.2 Код хөгжүүлэлт	33
4.3 Үр дүн	44
5. ДҮГНЭЛТ	49

НОМ ЗҮЙ	50
ХАВСРАЛТ	51
A. SEAMLESSM4T МОДЕЛ	51
B. CHIMEGE -Г API-ИАР АШИГЛАХ	52
C. FLUTTER ДЭЭР AWS-ДЭЭРХ СЕРВЕР ЛҮҮ ХАНДАХ	53
D. ХЭРЭГЛЭГЧИЙН AGORA TOKEN ҮҮСГЭХ ENDPOINT	54
E. ЧАТЫН МЭДЭГДЭЛ ИЛГЭЭХ ENDPOINT	55
F. ЧАТ ӨРӨӨ БҮРИЙН ХУВЬД ШИНЭ ЧАТЫГ СОНСОХ EVENT	56

Зургийн жагсаалт

Зураг	Хуудас
1 Интернет хэрэглэгчдээс авсан судалгааны үр дүн	2
1.1 WhatsApp аппын харагдах байдал	4
1.2 Telegram-н харагдах байдал	6
1.3 2019-2022 онд хамгийн ихээр ашиглаж буй мобайл фрэймворкийн жагсаалт	7
1.4 Agora дуудлагын тойм зураг	10
3.1 Client-Server Архитектурын үлгэр загварын дүрслэл	15
3.2 Хэрэгжүүлэх гэж буй системийн архитектур	16
3.3 Ажлын явцын диаграм	17
3.4 Persona 1 - Бямбасүхийн мэдээлэл	19
3.5 Persona 2 - Хүслэн	20
3.7 Чатын хуудас	22
3.11 Системийн RD Schema	27
4.1 Төслийн файлын бүтэц	32
4.2 JSON объектын харагдац	39
4.3 Нүүр хуудас	45
4.6 Чат хуудас	48

Хүснэгтийн жагсаалт

2.1	Функциональ шаардлага	12
2.2	Функциональ бус шаардлага	13
2.3	User Interface дизайны шаардлага	14
3.1	User хүснэгт	28
3.2	Users хүснэгт	28
3.3	Chatroom хүснэгт	29
3.4	Chat хүснэгт	30
3.5	Audio chat хүснэгт	30
3.6	Photo хүснэгт	30

Кодын жагсаалт

1.1	Flutter ашиглаж "Container" widget буцаах функц	7
4.1	Хэрэглэгчийн бусад хэрэглэгч нартай харилцсан каналууд харагдах хуудас	33
4.2	Өгөгдлийн сангаа бүтэцлэх мөн объект нэмэх frame	38
4.3	Flask дээрээ end point гаргах	39
4.4	Dio ашиглан хүсэлт илгээж өгөгдлөө татаж авах	42
4.5	FirebaseFirestore-ийн чат өрөөний чатууд руугаа орчуулсан аудио файлын URL-г илгээх	43

УДИРТГАЛ

Орчин цагт хүмүүс хоорондоо сошиал платформуудын тусламжтайгаар өөрсдийн цаг заваа алдалгүй хүссэн нэгэнтэйгээ шууд харилцах буюу чат бичих, видео дуудлага хийх зэрэг боломжтой болсон. Хүмүүсийн хоорондоо харилцах цар хүрээ улам л тэлж аажмаар олон улс хооронд уг платформын тусламжтайгаар харилцдаг болоод аль хэдийнээ эхэлжээ. Иймд өөр хэл соёлтой тухайн сошиал бүтээгдэхүүнийг хэрэглэж буй хэрэглэгчдийг хоорондоо ойлголцоход хялбар болгох шаардлага гарч байгаа юм.

Зорилго

Хүмүүсийн харилцааг хөнгөвчилдөг хэл солилцох мобайл аппликейшн бүтээх ба тухайн апп нь видео дуудлага болон чат үйлдлүүдийг хийдэг бөгөөд хэрэглэгчийн сонгосон хэлний дагуу тухайн чатыг текстээс текст, харин видеог дуудлагыг аудионоос аудио орчуулга хийдэг байхаар зорьж байна.

Зорилт

Уг мобайл аппыг хөгжүүлэхдээ дараах үе шатын дагуу ажиллана.

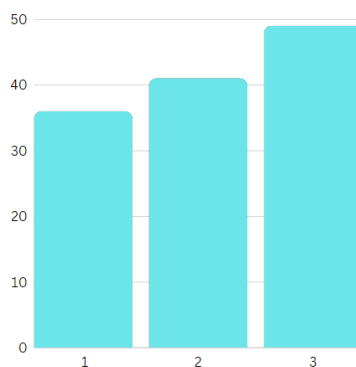
1. Хэрэглэгчийн үндсэн шаардлагуудыг тодорхойлох
2. UX судалгаа хийж хэрэглэгч суурьтай хялбар интерфейс дизайн гаргах
3. Ашиглах технологийг онол болон практик дээр суурилж судлах,
4. Системийн архитектурыг зохион байгуулж бэлдэх
5. Гаргасан баримт бичиг дээрээ тулгуурлаж хөгжүүлэлтээ хийх

Сэдэв сонгох үндэслэл

2023 оны 1-р сарын байдлаар сар тутамд 2 тэрбум хүн WhatsApp, 931 сая хүн мессенжерийг ашиглаж өдөр тутмын амьдралдаа бусадтай харилцах, мэдээ мэдээлэл солилцох

хэрэгцээгээ хангадаг байна.¹ . Өөрөөр хэлбэл дэлхийн хүн амын 1/4 нь сард хоорондоо уулзах, утсаар ярихаас гадна харилцаа холбооны сошиал платформуудыг ашигладаг гэсэн үг юм.

Миний хувьд сошиал сүлжээ ашиглан уг сэдэвтэй холбоотой судалгааг 53 интернэт хэрэглэгчдээс авсан бөгөөд тэдгээрт тулгарсан асуудлууд болон хариултуудыг дүгнэж үзвэл



Зураг 1: Интернэт хэрэглэгчдээс авсан судалгааны үр дүн

1. Чат бичихдээ Гүүгл орчуулгыг ашигладаг - 36 хүн
2. Гадаад хүнтэй видео дуудлага хийхэд орчуулагчтай байдаг - 41 хүн
3. Өөрсдийн сурахыг хүсч буй хэлээ өдөр тутам ашигладаг уг аппаар дамжуулан сайжруулах хүсэлтэй - 49 хүн

гэсэн хариу гарсан. Иймд эдгээр болон бусад хэрэглэгчдэд харилцах, хэлний асуудлаа шийдэхэд туслах боломжтой сошиал платформыг бүтээсэн нь зөв гэж үзсэн тул уг сэдвийг сонгон хөгжүүлж байна.

Ач холбогдол

Уг платформыг бүтээснээр миний хувьд бүтээгдэхүүнийг эхнээс нь дуусах хүртэлх хөгжүүлэлтийн үе шатуудтай танилцах, түүнийгээ практикаар хэрэгжүүлэх боломж бүр-

¹Дэлхийн мобайл мессенжэр апп хэрэглэгчдийн судалгаа: <https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/>

дэнэ.

Мөн бүтээгдэхүүн болгон гаргаж хэрэглээнд нэвтрүүлснээр дээр дурдсан асуудлыг тодорхой хэмжээнд багасгах боломжтой гэж үзэж байна. Олон зорилгоор хэрэглэх боломжтой ба жишээ нь өөрийн сурахыг хүсч буй хэлээрээ чат бичих, өөрийн хэлэхийг хүсч буй өгүүлбэр нь яаж бичигддэг мөн хэрхэн уншигддаг вэ зэргийг ойлгох цаашлаад хэл хамаарахгүй хэн нэгэнтэй харилцдаг болох боломж бүрдэх юм.

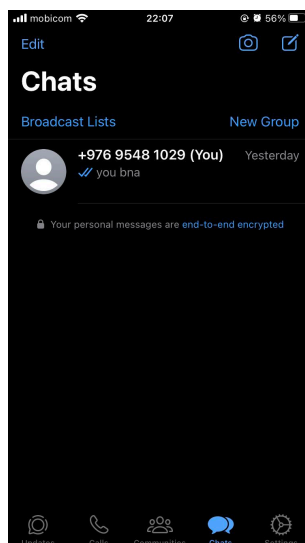
1. СЭДВИЙН ЕРӨНХИЙ СУДАЛГАА

Сэдвийн хүрээнд өмнө нь ажиллаж үзээгүй олон шинэ технологиудыг судалж, хооронд нь харьцуулан системдээ ашиглахад хамгийн тохиромжтой технологиудыг сонгох, цаашлаад технологийн цар хүрээнд тааруулж системээ хэрхэн өргөжүүлэх боломжтой талаарх мэдлэгийг авлаа.

1.1 Ижил төсөөтэй систем

1.1.1 *WhatsApp*

WhatsApp-ыг 2009 оны 2-р сард Yahoo!-ийн ажилтан асан Брайан Эктон, Ян Кум нар үүсгэн байгуулжээ. Өдгөө уг апп нь 180 гаруй улсад ашиглагддаг шилдэг харилцаа холбооны платформуудын нэг болсон байна.



Зураг 1.1: WhatsApp аппын харагдах байдал

Манай бүтээх гэж байгаа аппаас ялгаатай тал нь

- Speech-to speech орчуулга хийж болдоггүй
- Text-to-text орчуулга бүхий мессежийг үнэгүй хийдэггүй

1.1. ИЖИЛ ТӨСӨӨТЭЙ СИСТЕМ БҮЛЭГ 1. СЭДВИЙН ЕРӨНХИЙ СУДАЛГАА

- Интерфейсийн хувьд илүү олон мэдээлэлтэй

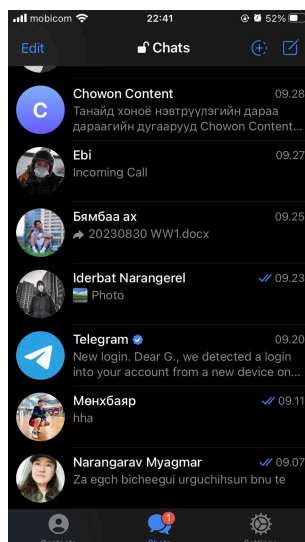
Уг апп нь ямар ч үйлдлийн систем дээр ажилладаг, VOIP дуудлага дэмжигддэг. Хөгжүүлэлтийн хувьд back-end нь Erlang програмын хэл дээр бичигдсэн харин Android үйлдлийн систем дээрх мобайл апп нь Жава болон Котлин дээр, IOS нь Swift ба Objective-C дээр бичигдсэн байна. Одоогоор whatsapp нь маш өргөн хэрэглээтэй апп бөгөөд үүний гол хүчин зүйл нь үнэгүй мессеж бичиж болдог ба үүлэн технологид суурилсан буюу хаанаас ч хандах боломжтой апп юм.

1.1.2 Telegram

Телеграммыг Оросын алдартай нийгмийн сүлжээний платформ болох ВКонтакте (VK) үүсгэн байгуулагч гэдгээрээ алдартай Николай Дуров, Павел Дуров нар үүсгэн байгуулжээ. Уг платформын хөгжил 2013 оны эхээр эхэлсэн бөгөөд ах дүү Дуров нар одоо байгаа мессежийн аппликешнүүдэд өгөгдлийн нууцлал, аюулгүй байдлын талаархи санаа зовоосон асуудлын хариуд аюулгүй, хувийн мессежийн платформыг бий болгохоор зорьсон байна. Өдгөө дэлхий даяар сар тутамд 931 сая идэвхтэй хэрэглэгчтэй хүчирхэг платформуудын нэг болж чаджээ.

Онцлог нь гэвэл Telegram нь дуут болон видео дуудлага ба үнэгүй мессеж илгээх зэрэг үйлдлүүдийг дэмждэг бөгөөд энэ нь хэрэглэгчдэд найз нөхөд, харилцагчидтайгаа өндөр чанартай харилцах боломжийг олгодог. Аюулгүй байдлын хувьд уг апп нь E2EE ашигладаг тул мэдээллийн аюулгүй байдлын хувьд найдаж болохоор үзүүлэлттэй ба аль ч платформуос үүлэн технологийн тусламжтайгаар өөрийн аккаунт руугаа нэвтэрч болох юм.

Уг аппын хөгжүүлэлтийн технологийн хувьд Back-End болох үндсэн логик үйлдлүүд нь Python дээр, мобайл апп нь React-native дээр хөгжүүлэгдсэн байна. Харин Веб хувилбар нь цэвэр HTML, CSS Javascript дээр хийгдсэн бөгөөд бодит хугацаанд видео болон аудио дамжуулахдаа WebRTC-г ашиглажээ.



Зураг 1.2: Telegram-н харагдах байдал

1.2 Ашиглах технологи

1.2.1 Flutter

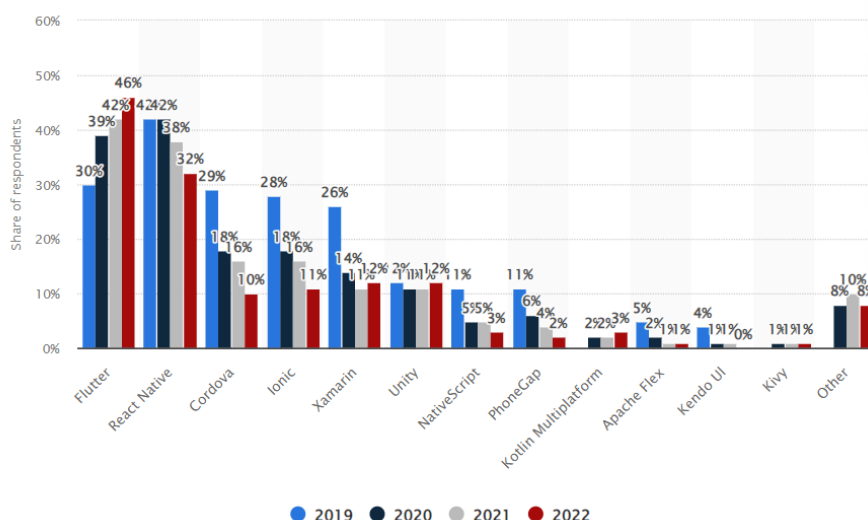
Сонгосон шалтгаан

Хэдийгээр олон төрлийн технологи ашиглан уг бүтээгдэхүүнийг хөгжүүлэх боломжтой ч орчин үед хамгийн трэнд болж буй уг технологийн тусгайлан сонгож аван судалж үзлээ. Хамгийн эхлээд 2019-2022 онд хамгийн ихээр ашиглагдаж буй мобайл фрэймворкийн судалгааг¹ танилцуулъя.

Энэ судалгаанаас та Flutter-г мобайл апп хөгжүүлэлтэд хамгийн ихээр ашиглаж буйг харж болно.

Мөн Dart хэл ашиглан хийсэн, хөгжүүлэлт хийхэд хялбар, client side аппликейшн хийх боломжтой мөн Widget гэсэн ойлголтыг оруулж ирснээр код бичихэд хялбар байдлыг олгосон нь уг хэрэгслийг ашиглах хамгийн том шалтгаанууд юм.

¹Мобайл фрэймворкийн судалгаа <https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/>



Зураг 1.3: 2019-2022 онд хамгийн ихээр ашиглаж буй мобайл фрэймворкийн жагсаалт

Технологийн талаар

Гүүгл компани дотооддоо ашиглаж байсан технологио 2018 онд танилцуулсан нь програмчлалын Dart хэлийг ашиглаж хийсэн програм хөгжүүлэлтийн фрэймворк болох Flutter² технологи юм. Үүнд widget ба түүний lifecycle мөн хамгийн чухал зүйл болох State management зэрэг багтана.

Flutter нь бүх үйлдлийн систем дээр ажиллаж болох фрэймворк бөгөөд апп дээрх объект бүрийг widget гэж явдаг тул олон нэр томъёг ойлгох шаардлагагүй хялбар хэрэгсэл юм. Widget үүсгэж бичихийн давуу тал нь нэг бичсэн кодоо олон дахин бичигдэхээс зайлсхийж, дахин ашиглах боломжийг олгодог үүгээрээ Component гэх ойлголттой төстэй. Мөн widget нь stateless, stateful гэж хоёр хуваагддаг ба stateful widget нь өөрийн гэсэн төлөвтэй, түүнийгээ удирддаг class болон state ашигласан функцууд байна. Flutter-н давуу тал нь state management-н тусламжтайгаар өөрчлөлтийг үргэлж хянаж байдаг тул өөрчлөлт орж ирэхэд бүтэн хуудсыг зурах бус зөвхөн тухайн өөрчлөгдсөн widget-г л дахин зурж чаддаг. Ингэснээр энгийн app-уудаас илүү хурдтай ажиллах боломжтой юм.

```
1 Widget Demo(String children) =>
2   return
```

²Flutter official site <https://flutter.dev/>

```
3 Container(child:Text(children));
```

Код 1.1: Flutter ашиглаж "Container" widget буцаах функц

Dart програмчлалын хэл нь Гүүгл компанийн хөгжүүлсэн хэл бөгөөд компьютерийн апп, мобайл апп ба вэб аппыг хөгжүүлэхэд ашиглагддаг байсан хэл юм. Хожим нь уг хэл дээр суурилан хөгжүүлэгдсэн технологи нь Flutter фреймворк ба уг хэрэгслийг ашиглан дан thread бүхий хурдан ажиллагаатай програмыг хийдэг байна.

1.2.2 Flask - Python дээр суурилсан фреймворк

Сонгосон шалтгаан

Flask нь python програмчлалын хэл дээр бичигдсэн микро фреймворк билээ. Мөн Flask нь RESTful API-г бүтээхэд маш тохиромжтой ба HTTP аргуудын дэмжлэг нь вэб үйлчилгээ болон API хөгжүүлэхэд түгээмэл сонголт болдог. Уг Фреймворк нь даалгавруудыг гүйцэтгэхэд гадны сангаас хамааралгүй байдаг тул олон хөгжүүлэгчид Flask-аас эхлэхийг илүүд үздэг.

Flask-н давуу талуудаас дурьдвал:

- Flask нь гадаад сангуудаас хамааралгүй бөгөөд нарийн төвөгтэй програмуудын хөгжүүлэлтийг хурдан эхлүүлэх боломжийг олгодог хөнгөн бүтэцтэй юм.
- Flask нь ML, agile хөгжүүлэлт, үүлэн технологи гэх мэт хамгийн сүүлийн үеийн технологитой нийцтэй ажилладаг.
- Энэ нь микрофреймворк учраас үндсэн үйлдлээ хөгжүүлэхэд хурдан байдаг.
- Flask нь дэбаг хийх, хөгжүүлэх, тест хийхэд нэгдсэн туршилтын функцийг санал болгодог.
- Flask нь вэб хөгжүүлэх үйл явцад уян хатан байдал, өргөтгөх чадварыг сайжруулахад тусалдаг WSGI загваруудыг дэмждэг гэх мэт маш олон давуу талуудтай

Мөн хиймэл оюунтай ажилладаг ихэнх функцүүд python хэл дээр бичигдсэн байдаг тул Flask шиг уян хатан хэрэгсэлийг ашиглах нь AI model integration талдаа илүү хялбар

боломжийг олгож өгдөг.

Технологийн талаар

Flask³ нь Python хэл дээр бичигдсэн фрэймворк юм. Үүнийг Поокко хэмээх олон улсын Python сонирхогчдын багийг ахалсан Армин Роначер боловсруулсан. Flask нь Werkzeug WSGI хэрэгсэл болон Jinja2 загвар хөдөлгүүр дээр суурилдаг ба энэ 2 нь 2 уулаа Поокко-ын төсөл юм. Мөн Flask нь python хэл дээр бичигдсэн микро вэб фреймворк буюу апп-ын гол үйлдлүүдийг хийхэд ашиглагддаг, хэрэглэхэд хялбар хэрэгсэл билээ. Тиймээс уг фрэймворк сүүлийн жилүүдэд шилдэг back-end service-үүдийг бичих сайн сонголтуудын нэг байсаар байгаа билээ.

1.2.3 FirebaseFirestore- Өгөгдлийн сан

Firebase нь үүлэн технологид суурилсан олон Back-End сервисүүдийг багтаадаг цогц Back-End үйлчилгээ бүхий платформ юм. Тэдгээр үйлчилгээнүүдийн нэг нь FirebaseFirestore өгөгдлийн сан ба өөрийн хийж буй аппынхаа хувьд уг NoSQL өгөгдлийн санг ашиглахаар сонгосон билээ. Мөн Firebase нь authentication буюу хэрэглэгчийн бүртгүүлэх, нэвтрэх зэрэгтэй холбоотой үйлчилгээнүүдийг дэмждэг бүтээгдэхүүн юм.

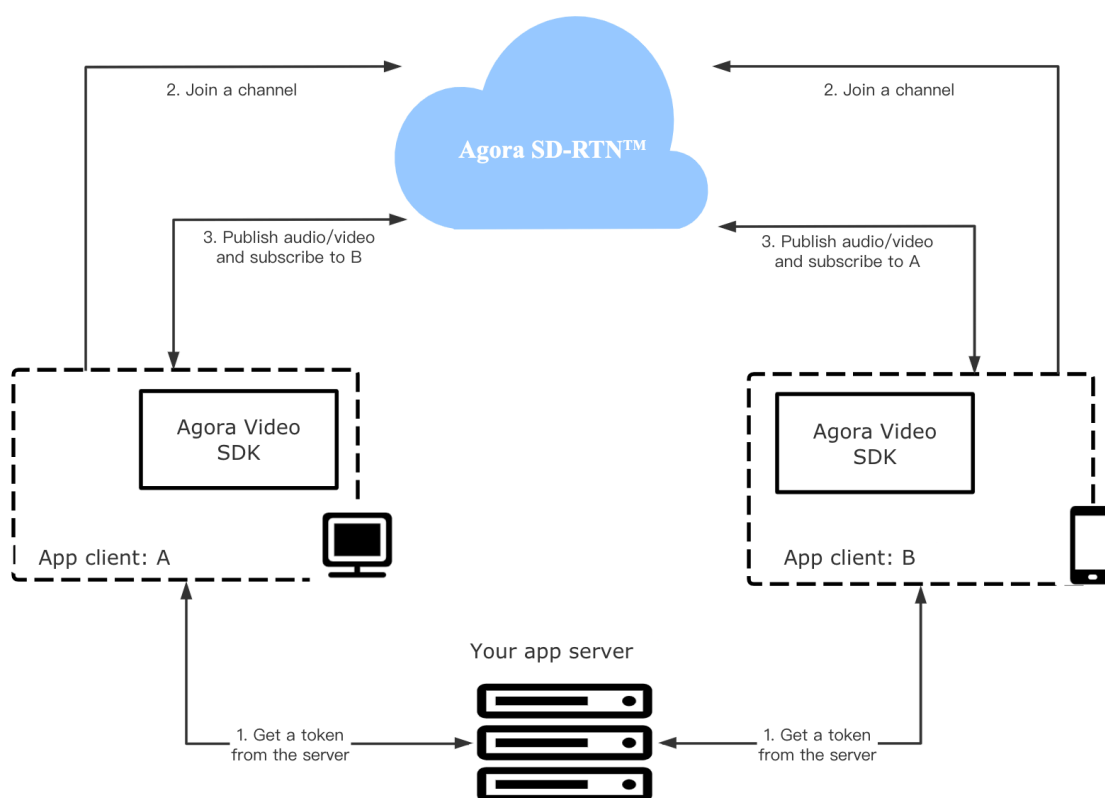
Түүхийн хувьд Firebase нь 2011 онд Жеймс Тамплин, Эндрю Ли нарын үүсгэн байгуулсан Envolvе start ап-аас хөгжсөн. Envolvе нь хөгжүүлэгчдэд онлайн чатын функцийг вэбсайтаа нэгтгэн ашиглах боломжийг олгодог API-аар хангадаг start-ап байсан юм. Чат үйлчилгээг гаргасны дараагаар Тамплин, Ли нар үүнийг чат мессеж биш програмын өгөгдлийг дамжуулахад ашиглаж байгааг олж мэдээд чатын систем болон бодит хугацааны өгөгдөл дамжуулах үйлчилгээнүүдийг салгахаар шийдэн 2012 оны 4-н сард бодит хугацааны өгөгдлийн сангаа Firebase нэртэй Real-Time өгөгдлийн сангийн үйлчилгээ үзүүлдэг компани болгон байгуулсан байна.

³Flask official site <https://flask.palletsprojects.com/en/3.0.x/>

1.2.4 Agora- Real time холболт үүсгэгч

Agora нь real time видео, аудио дуудлага болон чатлах үйлчилгээнүүдийг ашиглах боломжоор хангаж өгдөг бэлэн SDK тай платформ юм. Уг платформ нь түгээмэл ашиглагддаг 3 хэсгээс бүрдэх ба үүнд

1. Agora voice calling
2. Agora video calling
3. Broadcast Streaming



Зураг 1.4: Agora дуудлагын тойм зураг

Уг технологийг сонгох болсон гол шалтгаан нь Flutter дээр хөгжүүлэлт хийх боломжтой бөгөөд бэлэн SDK-ашиглан бодит хугацааны дуу, видео дамжуулах үйлчилгээг хэрэгжүүлэх боломжтой байсан. Энэ нь миний хувьд илүү хурдан хугацаанд хангалттай үр дүнд хүрэх боломжийг өгч байна.

1.2.5 Figma - интерфэйс дизайн, Prototype хувилбар гаргах багаж

Миний хувьд уг платформоо бүтээхдээ хөгжүүлэлтийн үе шатуудыг судлах, түүнийгээ практик дээр хэрэгжүүлэх зорилготой байгаа эхнээс нь бүхий л зүйлсээ чанартайгаар гүйцэтгэхийг оролдож байгаа. Уг үе шатуудад орчин үеийн аргачлалаар хэрэглэгчийн интерфэйс дизайн гаргах, түүнийгээ Prototype түвшинд хүргэж туршиж үзэх нь зайлшгүй багтана. Иймд User Experience болон хэрэглэгчийн интерфэйсээ гаргахдаа веб дээр суурилсан Figma гэх багажтай танилцан, ашиглаж байна.

Figma нь 2015 онд анх үүсэж байсан бөгөөд веб суурьтай тул ямар ч үйлдлийн систем дээр ажиллах боломжтой. Мөн Vector болон Raster төрлийн зурагтай харьцдаг, олон дизайнерууд бодит хугацаанд хамтдаа ашиглах боломжтой, веб болон гар утасны аппын интерфэйс хөгжүүлэлтэй ижил системээр (жишээ нь адилхан component гэх ойлголттой) явдаг нь үүнийг сонгох хамгийн том давуу тал болж өгсөн.

Зурсан интерфэйсүүдээ хооронд нь холбож хийсвэрээр аппаа ажиллуулан хэрэглэгчийн туршилт хийх хэсгийг Prototype гэдэг бөгөөд заавал кодын хэрэгжүүлэлт хийж цаг хугацаа болон мөнгөн зардал гаргалгүйгээр хийж буй аппаа хэрэглэгчээр туршуулах, үр дүнгээ гарган авч түүнийгээ сайжруулах нөхцөлийг уг веб аппликейшн маань гаргаж өгсөн нь UX/UI дизайнеруудын ашиглах болсон хамгийн том шалтгаануудын нэг юм.

2. СИСТЕМИЙН ШААРДЛАГА

Уг бүлэг нь системийн хэрэглэгчийн зүгээс тавигдах шаардлагыг тодорхойлж, тухайн гаргасан шаардлагууд дээрээ үндэслэн UX судалгаа хийсэн талаар гарах ба хэрэглэгч суурьтай интерфейс дизайн гаргахад тулгарсан асуудлуудыг товч дурдлаа.

2.1 Шаардлагын шинжилгээ

2.1.1 Хэрэглэгчид

Интернет сүлжээ ашиглан хэн нэгэнтэй харилцдаг,өөр хэл сурахыг хүссэн бүх төрлийн хэрэглэгчид

2.1.2 Функционал шаардлагууд

Хүснэгт 2.1: Функциональ шаардлага

ФШ 101	Хэрэглэгч системд мэйл хаягаа ашиглан бүртгүүлэх бас нэвтэрдэг байх
ФШ 102	Хэрэглэгч уг системд бүртгэлтэй дурын хэрэглэгчтэй харилцах боломжтой байх
ФШ 103	Хэрэглэгч тухайн сонгосон хүнтэйгээ видео дуудлага хийх боломжтой байх
ФШ 104	Хэрэглэгч тухайн сонгосон хүнтэйгээ чат бичих боломжтой байх
ФШ 105	Дуудлага хийх, чат бичихээсээ өмнө нь ямар хэлнээс ямар хэл рүү хэл солицохыг шийддэг байх
ФШ 106	Дуудлага хийх явцад хэрэглэгчийн ярьсан яриаг сонгосон хэлний дагуу нөгөө талд орчуулан хэлж өгдөг байх
ФШ 107	Чат бичих явцад хэрэглэгчийн бичсэн текстийг сонгосон хэлний дагуу солин нөгөө талд 2 хэл дээрх текстийг харуулдаг байх.

2.1.3 Функционал бус шаардлагууд

Хүснэгт 2.2: Функциональ бус шаардлага

ФБШ 101	Мобайл апп нь хэрэглэгч ашиглахад хялбар интерфейстэй байх
ФБШ 102	Интерфейс дизайн нь UI шаардлагын дагуу хөгжүүлэгдсэн байх
ФБШ 103	Видео дуудлагын үед нөгөө хэрэглэгчид орчуулсан буюу exchange хийсэн яриа нь 10 секунд дотор сонсогддог байх.
ФБШ 104	Чат бичих үед тухайн текст чатыг сонгосон хэлний дагуу солисон буюу орчуулсан текст нь 3 секунд дотор орчуулагдаад илгээгддэг байх
ФБШ 105	Чат бичих үед чат мэдэгдлийг оруулж өгөх
ФБШ 106	Систем нь хэрэглэгчийн хүссэн цагт ашиглагдахуйц боломжтой байх ёстой

2.2 UX/UI шаардлага

Уг судалгааны ажлын онцлог тал нь шууд хөгжүүлэлтээ хийж эхлэхээс өмнө хэрэглэгч төвтэй дизайн гаргаж түүнийгээ зорилтот хэрэглэгч дээр туршиж, гүйцэтгэл сайтай User Experience болон User Interface дизайн гаргах юм. Ингэснээр сайн бүтээгдэхүүн гаргах том үндэс болох, ирээдүйд гарах хөгжүүлэлтийн зардлыг багасгах давуу талтай.

2.2.1 User Experience шаардлага

Хэрэглэгч төвтэй UX/UI дизайн гаргахад дараах үе шатын дагуу ажиллах шаардлагатай.

- User Persona гаргах

Манай платформыг ашиглах боломжтой хоёроос дээш хэрэглэгчийг сонгож урьдчилж бэлдсэн асуултуудаас асууж мэдээллийг нэгтгэн тухайн хүнийг тодорхой хэмжээнд дүгнэн, уг платформыг ямар зорилготойгоор ашиглах боломжтой нөхцлүүдийг /Use Case/ гаргана.

- Асуудал тодорхойлох

Гаргасан Use Case дээрээ үндэслэж эцсийн хэрэглэгч дээр ямар асуудлууд байгааг судалж, хэрхэн шийдэх талаар санаа гарган User Experience-н шаардлагуудыг тодорхойлно. Энэ нь хэрэглэгч төвтэй дизайн гаргахын үндэс болох тул таамгаар бус судалгаан дээрээ үндэслэж вэбийн хэрэглэгчийн харилцах хэсгийн дизайныг гүйцэтгэнэ.

- Доод түвшинд Prototype хувилбар гаргах

Дээрх ажлуудыг нэгтгэн User Experience дизайныг доод түвшинд буюу ерөнхий загвартайгаар хийж, ашиглаж буй хэрэгсэл болох Figma дээрээ бүх хуудас, компонентийн логик үйлдлүүдийг холбож бодит ажилладаг мэт загвар гаргана.

- Usability туршилт хийж дизайнаа сайжруулах

Гаргасан Prototype хувилбараа сонгож авсан хэрэглэгчдээр ашиглиулж UX дээр ямар алдаа байгаа, хэрэглэгчдийн асуудлыг шийдвэрлэж чадаж байгаа эсэх, вэб компонентүүдийн байрлал, хоорондын логик үйлдэл зөв байгаа эсэхийг тодорхойлж хэрэглэгчдээс санал хүсэлт авах шаардлагатай. Түүний дараагаас дизайнаа дахин сайжруулж, хөгжүүлэлтийн шатанд ороход бэлэн болгох хэрэгтэй.

2.2.2 User Interface дизайны шаардлага

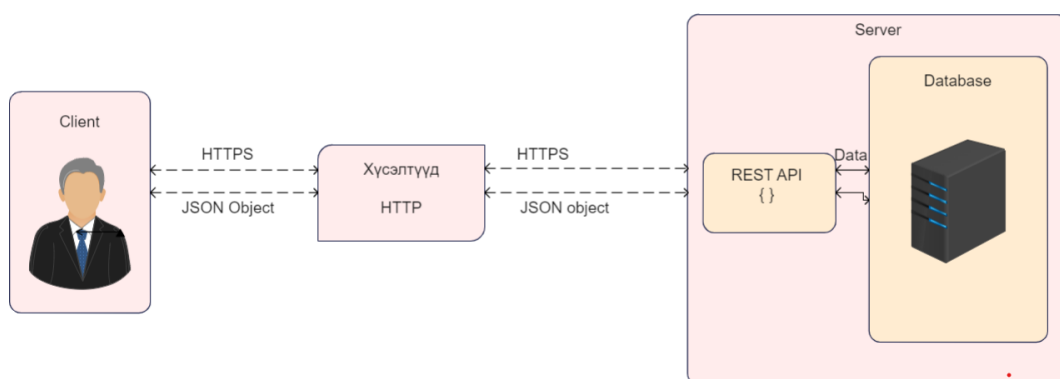
Хүснэгт 2.3: User Interface дизайны шаардлага

ИДШ 101	Телеграмтай төстэй дизайнтай байх
ИДШ 102	Гол элементүүд дээр "0057ff"hex кодтой өнгийг ашиглах
ИДШ 103	Компонентуудын micro-interaction ойлгомжтой байх
ИДШ 104	Дэвсгэр өнгө дээр цагаан өнгийг түлхүү ашиглах
ИДШ 105	Contrast буюу өнгөний ялгарлыг бага байлгах
ИДШ 106	Аппын фонтын хувьд "Nunito - Cyrillic Extended"хувилбарыг ашигласан байх
ИДШ 107	Элемент хооронд white-space буюу сул зайг сайн гаргаж өгөх

3. СИСТЕМИЙН АРХИТЕКТУР, ЗОХИОМЖ

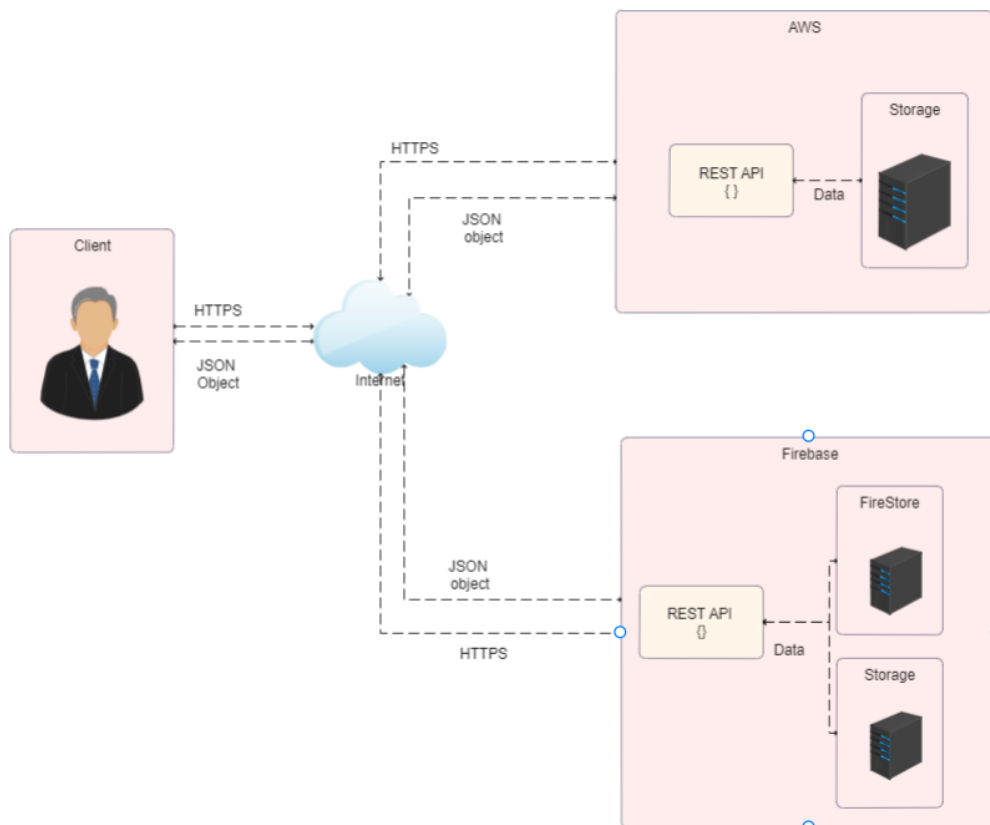
3.1 Системийн архитектур

Манай систем нь тийм ч төвөгтэй систем биш бөгөөд бүх төрлийн үйлдлийн системтэй утсанд зориулан хийгдэх тул системийн архитектурын үлгэр загвар дундаас мобайл аппуудад хамгийн өргөн хэрэглэгддэг Хэрэглэгч-Сервер (Client-Server pattern) үлгэр загварыг сонголоо. Хэрэгжүүлэхэд хялбараас гадна Flutter, AWS EC2, Flask, Firebase ашиглан төслөө хөгжүүлж байгаа учир хамгийн тохиромжтой гэж үзэж байна.



Зураг 3.1: Client-Server Архитектурын үлгэр загварын дүрслэл

Уг архитектур нь хэрэглэгч талаас гар утасны аппыг ашиглаж HTTP холболтоор хүсэлт явуулж, сервер талаас өгөгдлийн санг ашиглан REST API бэлдэн эргээд хэрэглэгч рүү HTTP холболтоор хүсэлтийн хариуг өгөх зарчмаар ажиллана.



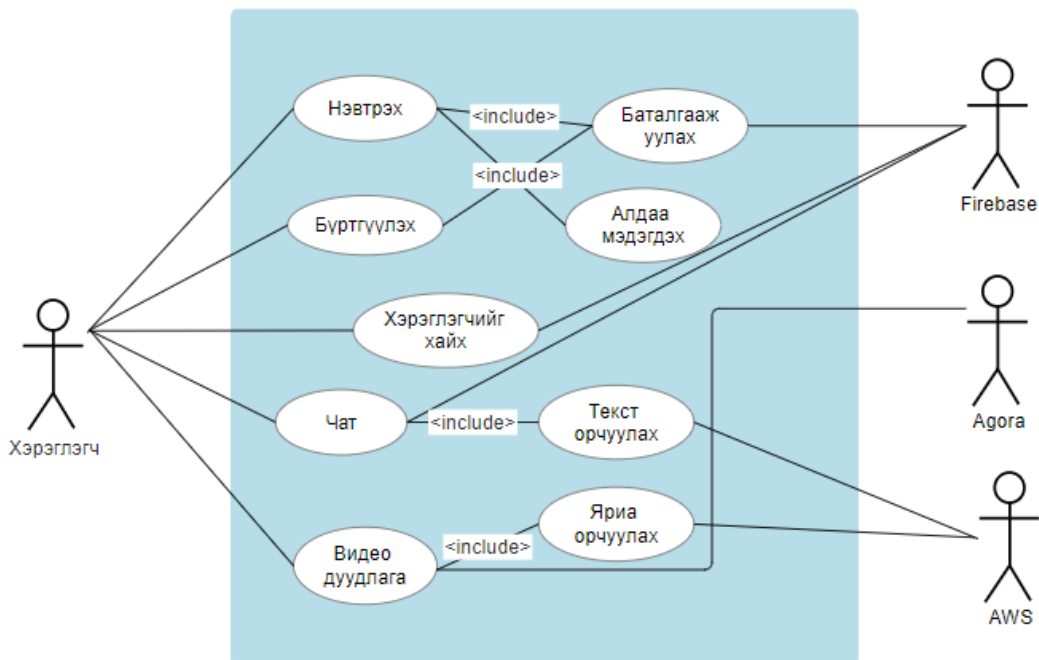
Зураг 3.2: Хэрэгжүүлэх гэж буй системийн архитектур

Front-end хэсгээ Flutter дээр, харин Back-end хэсгээ Flask дээрээ хийх буюу fullstack хөгжүүлэлт хийгдэнэ.

- Client талаас сервер рүү интернэт ашиглан HTTPS холболтоор өгөгдлөө дамжуулна
- AWS сервер дээр Flask-аар бичигдсэн API сервисийг ашиглан аудио болон текстээ хөрвүүлнэ.
- Хэрэглэгч нь хөрвүүлсэн аудио файлын URL-ыг авч FirebaseFirestore сервер луу мессэж болгон илгээнэ.

3.2 Ажлын явцын диаграм

3.2.1 Ажлын явцын диаграм



Зураг 3.3: Ажлын явцын диаграм

Ажлын явцын диаграммын тайлбар

- **Нэвтрэх** - Бүртгэлтэй хэрэглэгч өөрийн бүртгүүлсэн имэйл хаягаараа нэвтрэх, нууц үгээ сэргээх
- **Баталгаажуулах**- Системд нэвтрэх эсвэл бүртгүүлэхэд хэрэглэгчийг баталгаажуулаад нэвтрүүлэх
- **Алдааг мэдээлэх** - Хэрэглэгчийн оруулсан мэдээлэл буруу эсвэл алдаатай формат зэрэгтэй байвал алдааг мэдээллэдэг байх
- **Хэрэглэгчийг хайх** - Систем бүртгэлтэй хэрэглэгчдийг мэйл хаягаар нь хайн харилцаа үүсгэх боломжтой байх

3.2. АЖЛЫН ЯВЦЫН ДИАГРАММ 3. СИСТЕМИЙН АРХИТЕКТУР, ЗОХИОМЖ

- **Чат бичих** - Хэрэглэгч нь системд бүртгэлтэй ба өөрийн хүссэн хэрэглэгчтэйгээ бодит хугацааны чат бичих болдог байх
- **Видео дуудлага хийх** - Хэрэглэгч нь системд бүртгэлтэй ба өөрийн хүссэн хэрэглэгчтэйгээ бодит хугацааны видео дуудлага хийж болдог байх
- **Бүртгүүлэх** - Хэрэглэгч бүртгэлгүй хэдий ч манай платформыг бүрэн ашиглах боломжтой. Хэрэв өөрөө аппыг ашиглахыг хүсвэл хувийн мэдээллээ бөглөж бүртгүүлэх
- **Текст орчуулах** - Бүртгэлтэй хэрэглэгчдийн текст чатыг сонгосон хэлнийх нь дагуу орчуулах
- **Аудио орчуулах** - Бүртгэлтэй хэрэглэгчдийн видео дуудлага дахь ярьсан аудиог сонгосон хэлнийх нь дагуу орчуулах
- **Гарах** - Системд бүртгэлтэй хэрэглэгч системээс гарч болдог байна

3.3 UX/UI дизайн

Өмнөх бүлэгт хийсэн UX судалгаан дээрээ үндэслэж User Experience болон User Interface загварыг High Fidelity түвшинд эцэслэж гаргасан ба гаргасан дизайнаа хэрэглэгчээр туршиулж тодорхой үр дүнгүүдэд хүрч чадсан. Үүнд:

- Хэрэглэгчийн шаардлагыг бүрэн ойлгож, хэрэглэгч төвтэй дизайн гаргасан
- Тусдаа бүлэг ажил болгон хийснээр интерфейс загвартаа анхаарч шинэлэг, ашиглахад хялбар хэрэглэгчийн интерфейс загвар зурсан
- Хөгжүүлэлтийн шатнаас өмнө бүх хэрэглэгчийн интерфейсүүд дизайн системийн дагуу гарч дууссан тул front-end хөгжүүлэлтийн явцыг ихээр хурдлуулсан
- Интерфейсийн дагуу front-end код явагдах тул back-end код дээр dummy датагаар хөгжүүлж явах, төлөвлөх үе шат хялбар болсон. Front-end хэсэгтэй зөрөх магадлал багассан гэж ойлгож болно.

Одоо UX шаардлагын дагуу ажилласан процессоо богиноор тайлбарлая.

3.3.1 User Persona тодорхойлж эхний загварыг бүтээсэн нь

Шаардлагын дагуу хоёр хүнийг сонгон авч тэдгээр хүмүүсээс хэрэгтэй мэдээллүүдээ нэгтгэсэн ба доорхи зургаар илэрхийллээ.

User Persona 1 - Бямбасүх

Нэр: Бямбасүх	Байршил: Монгол улс, Улаанбаатар хот
Нас: 23	Боловсрол: ШУТИС, Холбооны анги, 2020
Хүйс: Эрэгтэй	Гэр бүл: Ганц бие
Мэргэжил: Холбооны инженер	Зан төлөв: Нээлттэй харилцаатай

Зураг 3.4: Persona 1 - Бямбасүхийн мэдээлэл

Use Case

- Өглөө ажил дээрээ гадаад харилцагч нар луугаа мэйл бичих шаардлага гардаг ба заримдаа grammarly AI гэх мэт хэрэгслүүдийг ашигладаг
- Заримдаа огт хэлийг нь мэдэхгүй гадаад хүнтэй ярих шаардлага гарвал шууд дууг нь орчуулж болдоггүй юм байхдаа гэж бодогддог.

User Persona 2 - Хүслэнгийн мэдээлэл

Нэр: Хүслэн	Байршил: Монгол улс, Улаанбаатар хот
Нас: 21	Боловсрол: ШУТИС, График дизайн, 2018
Хүйс: Эрэгтэй	Гэр бүл: Ганц бие
Мэргэжил: Дизайнер	Зан төлөв: Нээлттэй харилцаатай

Зураг 3.5: Persona 2 - Хүслэн

Use Case

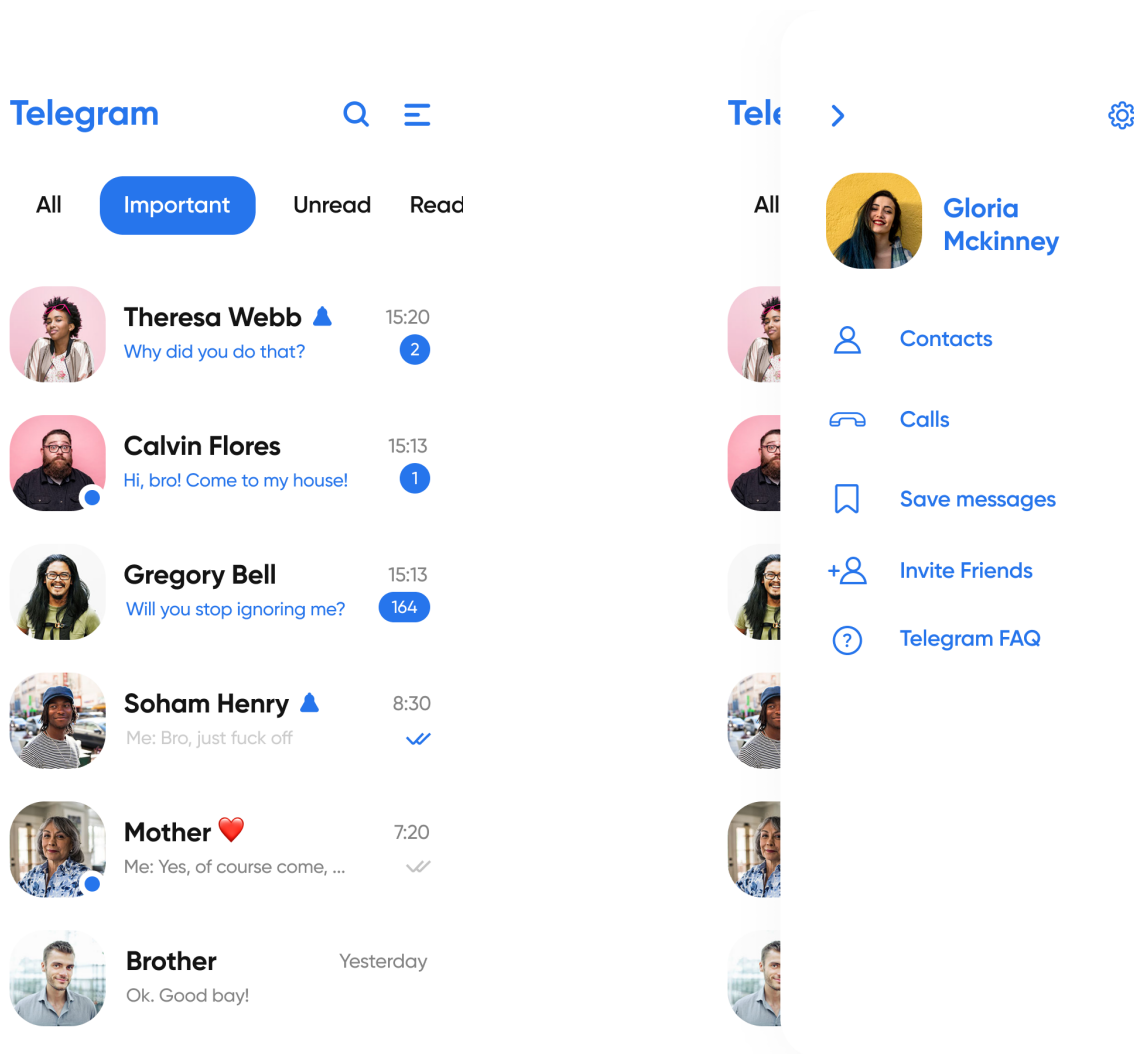
- Миний хувьд гадны өөр хэл соёлтой дизайнаруудтай харилцаж шинэ зүйл хийхэд бэрхшээл гардаг. Заримдаа видео дуудлага хийгээд ярих үе ч цөөнгүй тохионо энэ үед хэлний асуудал ихээр тулгардаг.
- Гадагшаа явж сурах сонирхолтой тул шинэ гадаад хэлүүд сурахыг хүсдэг. Гэвч их ажилтай байдаг болохоор зав гардаггүй. Тиймээс өдөр тутмын харилцаандаа өөр хэлээр харилцахыг их хичээдэг.

Эхний Wireframe загварыг гаргаж Prototype хувилбар бүтээсэн нь

Хамгийн эхэнд нийт 8 хуудсын Wireframe загварыг хэрэглэгчийн Use Case дээрээ тулгуурлан зурж, дизайн дээрх бүх элементүүдээ хооронд нь холбон Prototype ¹ хувилбар гаргаж, сонгож авсан хоёр хэрэглэгч дээрээ Usability туршилт хийхэд бэлэн болголоо.

¹Prototype хувилбарыг туршиж үзэх <https://www.figma.com/proto/EL2nOGmToBRVxHNuZwH661/Draft?>

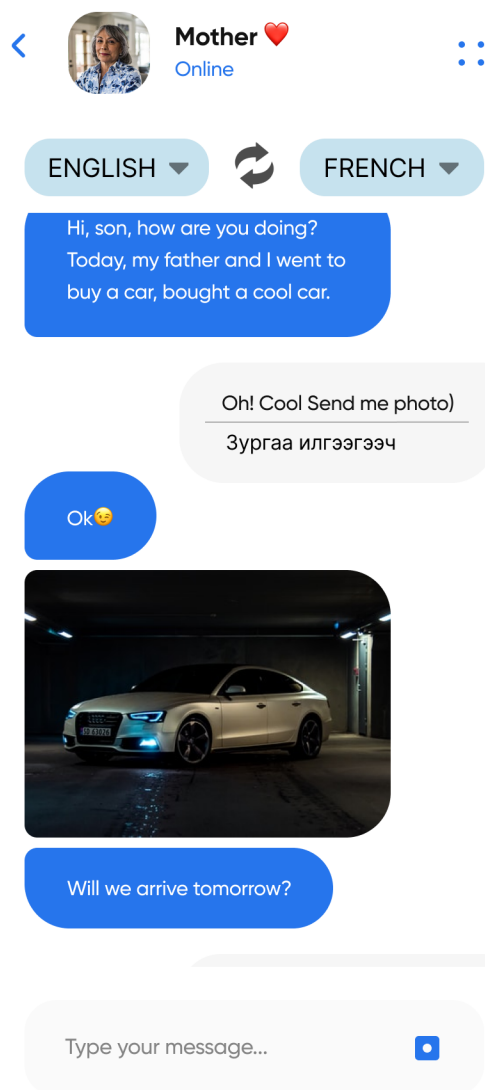
Нийт зурсан 8 хуудсаас вебийн гол процессийг илэрхийлэх 3 зургийг орууллаа. Бусад хэсгийг хавсралтаас ² үзэх боломжтой



(a) Үндсэн хуудас

(b) Хэрэглэгчийн мэдээлэл бүхий хэсэг

²Зурсан интерфейс дизайн <https://www.figma.com/file/EL2nOGmToBRVxHNuZwH661>



Зураг 3.7: Чатын хуудас

Prototype хувилбар дээрээ Usability туршилт хийсэн нь

Дараа нь Prototype хувилбар дээрээ Usability буюу хэрэглэхэд тухтай байдлын туршилтыг сонгож авсан хоёр хэрэглэгч дээрээ хийлгэж уг гаргасан дизайн дээрх үүссэн асуудлыг тодорхойлж, хэрэглэгчээс санал хүсэлтийг аван дараагийн гаргах дизайныхаа ерөнхий шаардлагуудыг тодорхойлов. Үүнээс дурдвал:

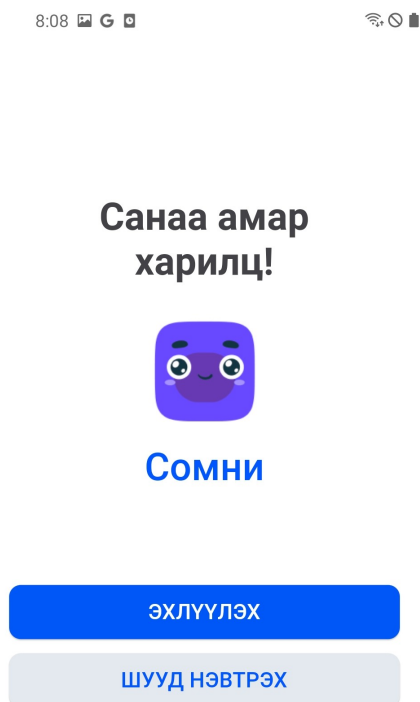
- UI загварын хувьд концептийг дахин сайжруулах
- Хэрэглэгчийг систем бүртгэхдээ түүний ярьж чаддаг хэлнүүдийг бүртгэх
- Хэрэглэгч рүү бичих чат эсвэл дуудлагынхаа орчуулах хэлийг нөгөө хэрэглэгчийн ойлгож чадах боломжит хэлнүүдээс сонгодог байх
- Дуудлагын түүхийг оруулж ирэх ба үүнийг доод талын ТабБар үүсгэн тухайн барын нэг таб болгон оруулж өгөх

Иймд уг шаардлага дээрээ үндэслэн эцсийн байдлаар аппынхаа UX/UI дизайныг гаргах шаардлагатай.

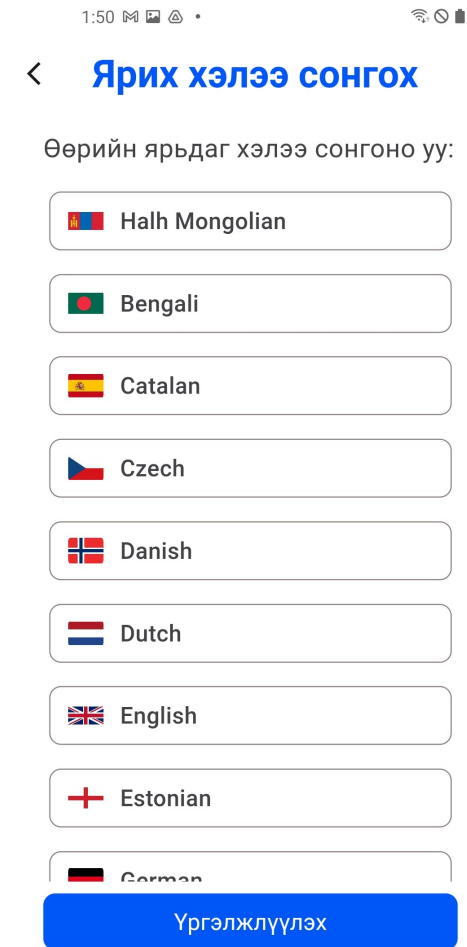
Redesign буюу зурсан дизайнаа дахин сайжруулсан нь

UX/UI гүйцэтгэлийн хамгийн сүүлийн шатанд нийт 10 хуудас дизайныг³ холбож зурсан ба өмнөх дизайны процесстой адилаар Prototype хувилбар гаргаж эцсийн хэрэглэгч дээрээ Usability туршилтыг хийж дараагийн шат болох код хөгжүүлэлтийн шатыг эхлүүлэхэд бэлэн боллоо. Нийт зурсан хуудсаасаа 6 хуудсыг онцлон харуулж, User Experience дизайныхаа шийдлийг тайлбарлалаа. Бусад зурсан хуудсыг доор байгаа хавсралтаас харах боломжтой.

³UX/UI дизайны сүүлийн хувилбар <https://www.figma.com/file/gDTk0c76R28sWuB9ey5gg7/Untitled?type=design&node-id=0-1&mode=design&t=uBIpYVqq1uh89a6M-0>




(a) Нүүр хуудас



(b) Хэл сонгох хуудас

8:07

←



Нэр

Имэйл

Нууц үг

Нууц үг баталгаажуулах


Бүртгүүлэх

This is a mobile app registration screen. It features a purple cartoon character at the top. Below it are four input fields: 'Нэр' (Name), 'Имэйл' (Email), 'Нууц үг' (Password), and 'Нууц үг баталгаажуулах' (Confirm Password). A blue button labeled 'Бүртгүүлэх' (Register) is at the bottom.

(a) Бүртгүүлэх хуудас

8:08

←



Имэйл

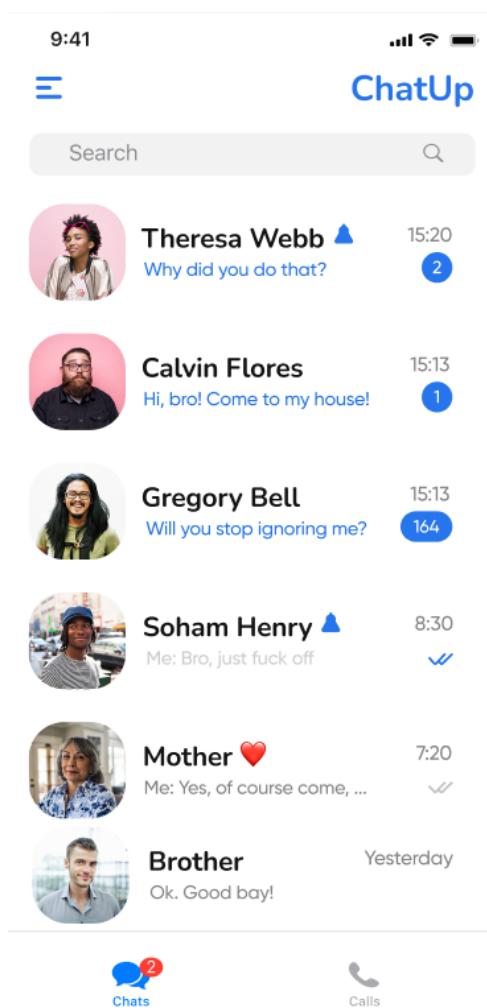
Нууц үг

Нэвтрэх

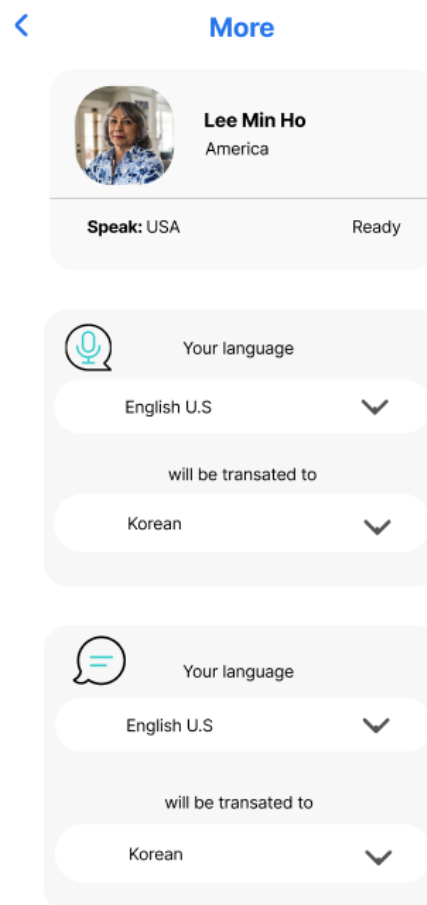
Forgot Password?

This is a mobile app login screen. It features a purple cartoon character at the top. Below it are two input fields: 'Имэйл' (Email) and 'Нууц үг' (Password). A blue button labeled 'Нэвтрэх' (Login) is at the bottom. Below the button is a link labeled 'Forgot Password?'. A red exclamation mark icon is visible next to the 'Имэйл' field.

(b) Нэвтрэх хуудас



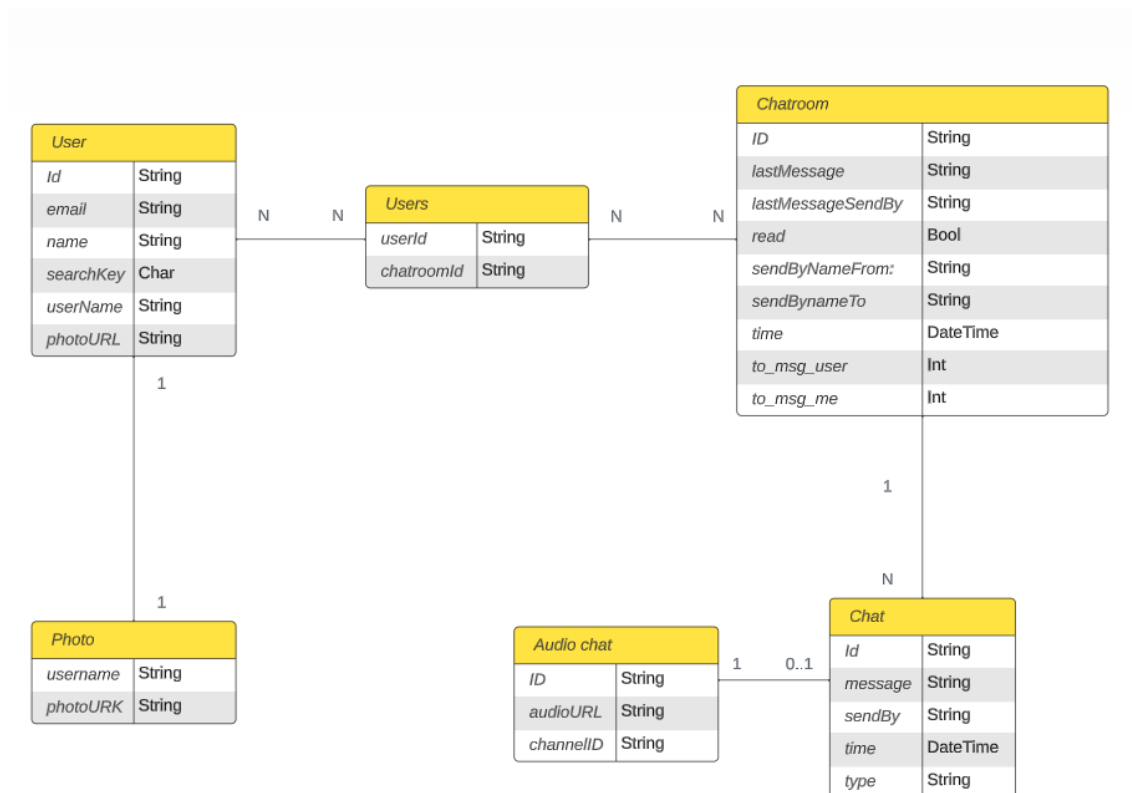
(a) Чат таб хуудас



(b) Чат болон дуудлагын хэлээ сонгох хуудас

3.4 Өгөгдлийн сан

3.4.1 Өгөгдлийн сангийн диаграм



Зураг 3.11: Системийн RD Schema

Өгөгдлийн сангийн хүснэгтүүдийн тайлбар

Хүснэгт 3.1: User хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	id	String	Хэрэглэгчийн дахин давтагдашгүй ID-г хадгална
2	Email	String	Хэрэглэгчийн гараас оруулж өгсөн имэйлийг хадгална. Зөвхөн латин тэмдэгтүүд ашиглах хэрэгтэй
3	Name	String	Хэрэглэгчийг өгөгдлийн сангаас хайх түлхүүр үг
4	userName	String	Хэрэглэгчийн системийн үйлдлүүдэд ашиглах unique нэр
5	photoURL	String	Хэрэглэгчийн өгөгдлийн санд оруулсан зурагны URL хаягийг хадгалах

Хүснэгт 3.2: Users хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	userId	String	Хэрэглэгчийн дахин давтагдашгүй ID-г хадгална
2	chatroomId	String	Хэрэглэгчийн бусад харилцагч нартай үүсгэсэн чат өрөө бүрийн дахин давтагдашгүй ID

Хүснэгт 3.3: Chatroom хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	id	String	Чатын дахин давтагдашгүй утгыг хадгална
2	lastMessage	String	Тухайн чат өрөөний хамгийн сүүлд бичсэн чатын мэдээллийг хадгална.
3	read	bool	Хамгийн сүүлд бичигдсэн чат уншигдсан уу эсвэл үгүй юу гэдгийг хадгална
4	sendByFromName	String	Тухайн чат өрөөн дахь хамгийн сүүлд бичигдсэн чат хэнээс илгээгдсэн бэ гэдэг мэдээллийг хадгална
5	sendByFromTo	String	Тухайн чат өрөөн дахь хамгийн сүүлд бичигдсэн чат хэн рүү илгээгдсэн бэ гэдэг мэдээллийг хадгална
6	time	DateTime	Тухайн чат өрөөн дахь сүүлийн чатын илгээгдсэн хугацааг хадгална
7	to-msg-user	int	Чат өрөө бүрийн хувьд тухайн хэрэглэгчийн илгээсэн уншигдаагүй чатын тоо
8	to-msg-me	int	Чат өрөө бүрийн хувьд тухайн хэрэглэгч рүү илгээгдэж ирсэн уншигдаагүй чатын тоо

Хүснэгт 3.4: Chat хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	id	String	Чат бүр өөрийн unique ID-тай байх ба чат бүр нь Чат өрөөнд хамаарагдана
2	message	String	Чат бүр нь өөрийн текстийн утгыг хадгална
3	sendBy	String	Чат бүрд тухайн чатыг илгээсэн хэрэглэгчийн мэдээллийг хадгална
4	time	DateTime	Тухайн чатын илгээгдсэн огноог хадгална
5	type	String	Тухайн чатын төрлийн мэдээллийг хадгална

Хүснэгт 3.5: Audio chat хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	ID	String	Аудио чатын unique ID-г хадгална
2	Audio URL	String	Тухайн дуут чатны сервер дээр хадгалагдсан URL хадгалах талбар
3	ChannelId	String	Тухайн дуут чатны хамаарагдах өрөөний мэдээллийг хадгалах талбар

Хүснэгт 3.6: Photo хүснэгт

№	Талбарын нэр	Өгөгдлийн төрөл	Тайлбар
1	username	String	Хэрэглэгчийн unique хадгалагдах нэрний талбар
2	photoURL	String	Хэрэглэгчийн зурагийн URL хаягийг хадгалах талбар

4. ХЭРЭГЖҮҮЛЭЛТ

Энэхүү бүлэгт өмнө нь гарсан интерфэйс дизайн, системийн архитектур болон диаграмуудыг хэрэгжүүлж хэрхэн бүтээгдэхүүн болгон гаргасан талаараа бичлээ. Хэрэгжүүлэлт хийх үе шатаа ерөнхийд нь

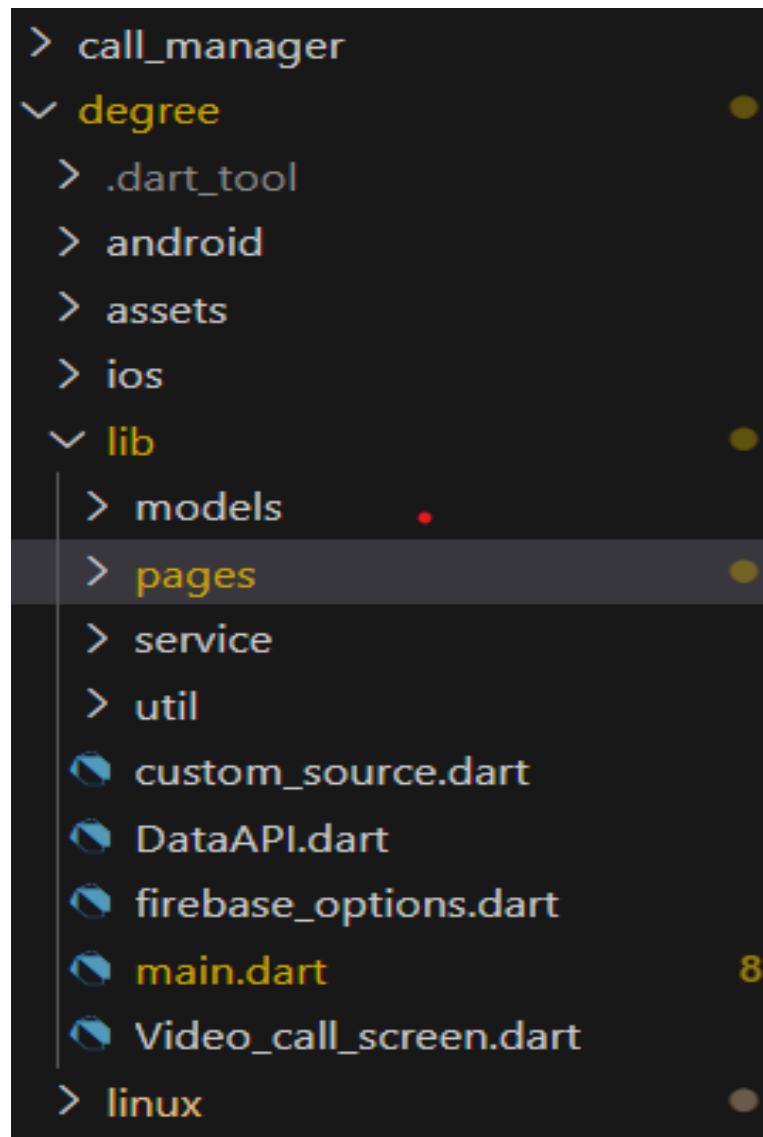
- Хөгжүүлэлтийн орчноо бэлдэх
- Front-end хөгжүүлэлт
- Back-end хөгжүүлэлт

гэсэн алхмуудад хувааж гүйцэтгэсэн ба доор ажлуудаасаа гол гэсэн зүйлсээ нэгтгэн оруулав.

4.1 Хөгжүүлэлтийн орчныг бэлдэх

Ашиглах технологийн хувьд Flask, Flutter, AWS, Firebase ашиглах учир хамгийн эхлээд FirebaseFirestore өгөгдлийн сангаа үүсгээд дараа нь Flutter фрэймворкоо суулган шаардлагатай тохиргоонуудыг хийнэ. Үүний дараагаас аппынхаа Back end -ийг хийж, үүнийгээ AWS EC2 дээр байршуулах юм.

Мөн кодын хувилбараа хадгалж, хөгжүүлэлтийн үе шатаа цэгцтэй авч явахын тулд Version Control дээрээ Git, Github.com-г сонгож public харагдацтай repository үүсгэв.



Зураг 4.1: Төслийн файлын бүтэц

Flutter дээр файлын бүтцээ Зураг 4.1 дээр харагдаж байгаачлан бүтэцлэж хөгжүүлэлтээ эхлэхэд бэлэн болголоо. Хөгжүүлэлтийн гол зорилго маань Fullstack хөгжүүлэлт хийх бөгөөд манай платформын back-end, front-end хэсгүүд тусдаа repository дээр явагдана. Front-end хэсгийн Файлын бүтцээ тайлбарлавал

- **model** хавтаст програмд ашиглагдах нэгжүүд(Entities) байршина
- **pages** хавтаст төслийн хуудаснуудын код файлууд байршина
- **service** хавтаст төслийн хуудаснуудад ашиглагдах Controller файлууд байршина

- **util** хавтаст төслийн Firebase-д шаардлагатай тохиргоо болон локал өгөгдлийн санг ашиглахад шаардлагатай глобал өгөгдлүүд байгаа.

4.2 Код хөгжүүлэлт

Хөгжүүлэлтийн хувьд маш олон компонент, хуудсууд, тэдгээрийн код хийгдсэн тул зөвхөн чат таб хуудас буюу хэрэглэгчийг нэвтэрсний дараах харагддаг хэсгийг онцлон авч эхнээс нь эхлээд бүтэн ажиллах хүртэлх үе шатуудыг тайлбарлав.

4.2.1 Гаргасан интерфэйсийнхээ дагуу чат таб хуудсыг өрсөн нь

Front-end хөгжүүлэлт дээр урьдчилж интерфэйс дизайныг гаргасан нь мобайл аппын бүх компонент, тэдгээр хаана байрших гээд олон зүйлийг тодорхой болгож өгсөн нь давуу тал болсон. Иймд хамгийн эхлээд бэлдсэн хөгжүүлэлтийн орчин дээрээ үндсэн хуудсууд болон дотор нь ашиглаж буй компонентуудыг статик байдлаар өрөв.

```
1
2  @override
3  Widget build(BuildContext context) {
4    textEditingController.value = textEditingController.value.copyWith(
5
6      selection:
7        TextSelection.collapsed(offset: textEditingController.text.length
8          ),
9    );
10   return Scaffold(
11     backgroundColor: Colors.white,
12     key: _globalKey,
13     appBar: HomeAppBar(),
14     drawer: DrawerBuilder(myName ?? _dataController.myname),
15     body: CustomScrollView(slivers: [
16       SliverAppBar(
17         elevation: 0,
18         floating: true,
19         titleSpacing: 0,
```

```
19         automaticallyImplyLeading: false,
20         title: Container(
21             // height: 35,
22             decoration: BoxDecoration(
23                 color: Colors.white,
24                 borderRadius: BorderRadius.all(Radius.circular(10)),
25             ),
26             margin: EdgeInsets.fromLTRB(20, 0, 20, 0),
27             child: TextField(
28                 cursorHeight: 25,
29                 controller: textEditingController,
30                 textAlignVertical: TextAlignVertical.center,
31                 focusNode: _focusNode,
32                 textAlign: search ? TextAlign.start : TextAlign.center,
33                 autocorrect: true,
34                 textCapitalization: TextCapitalization.words,
35                 onChanged: (value) {
36                     initiateSearch(value.toUpperCase());
37                 },
38                 decoration: InputDecoration(
39                     filled: true,
40                     fillColor: Color.fromARGB(255, 205, 205, 206),
41                     contentPadding: EdgeInsets.fromLTRB(10, 0, 0, 5),
42                     suffixIcon: IconButton(
43                         padding: EdgeInsets.fromLTRB(0, 0, 0, 0),
44                         icon: search
45                             ? GestureDetector(
46                                 onTap: () {
47                                     textEditingController.clear();
48                                     FocusScope.of(context).requestFocus(FocusNode());
49                                 },
50                                 search = false;
51
```

```
52         tempSearchStore = [];  
53         print('search');  
54  
55         setState(() {});  
56     },  
57     child: Icon(  
58         size: 25,  
59         Icons.close,  
60         color: Color(0Xff2675EC),  
61     ))  
62     : GestureDetector(  
63         onTap: () {  
64             search = true;  
65             _focusNode.requestFocus();  
66             print('not search');  
67             setState(() {});  
68         },  
69         child: Icon(  
70             size: 30,  
71             Icons.search,  
72             color: Color(0Xff2675EC),  
73         ),  
74     ),  
75     onPressed: () {  
76         search = true;  
77         setState(() {});  
78     },  
79 ),  
80 border: OutlineInputBorder(borderSide: BorderSide.none),  
81 enabledBorder: OutlineInputBorder(  
82     borderRadius: BorderRadius.circular(15.0),  
83     borderSide: BorderSide(  
84         color: Color.fromARGB(255, 205, 205, 206),  
85     ),
```

```

86         ),
87         focusedBorder: OutlineInputBorder(
88             borderRadius: BorderRadius.circular(15.0),
89             borderSide: BorderSide(
90                 color: Color.fromARGB(255, 205, 205,
91                     206),
92             ),
93         ),
94         hintText: 'Хайх',
95         hintStyle: const TextStyle(
96             fontFamily: "SF Pro Text",
97             fontSize: 16,
98             fontWeight: FontWeight.w400,
99             color: Color(0xff3c3c43),
100
101         ),
102     ),
103     style: TextStyle(
104         decoration: TextDecoration.none,
105         color: Colors.black,
106         fontFamily: 'Nunito',
107         fontSize: 15.0,
108         fontWeight: FontWeight.w400),
109     ),
110 ),
111 ),
112 Obx(() {
113     if (_dataController.roomsLen == 0)
114         return SliverToBoxAdapter(
115             child: Container(
116                 height: MediaQuery.sizeOf(context).height * 2 / 3,
117                 decoration: BoxDecoration(border: Border.all()),
118                 child: Center(child: Text('no item'))),
119         );

```

```

120         else
121             return SliverPadding(
122                 padding: EdgeInsets.only(top: 5),
123                 sliver: SliverList(
124                     delegate: SliverChildBuilderDelegate(
125                         (BuildContext context, int index) {
126                             // Build the list of items
127
128                             return search
129                                 ? ListView(
130                                     padding: EdgeInsets.only(left: 5.0, right:
131                                         5.0),
132                                     primary: false,
133                                     shrinkWrap: true,
134                                     children: [...tempSearchStore].map((element)
135                                         {
136                                             return buildResultCard(element);
137                                         }).toList())
138                                     : ChatRoomList();
139                                 },
140                                 childCount: 1,
141                             ),
142                         ),
143                     ));
144     }

```

Код 4.1: Хэрэглэгчийн бусад хэрэглэгч нартай харилцсан
чаннелууд харагдах хуудас

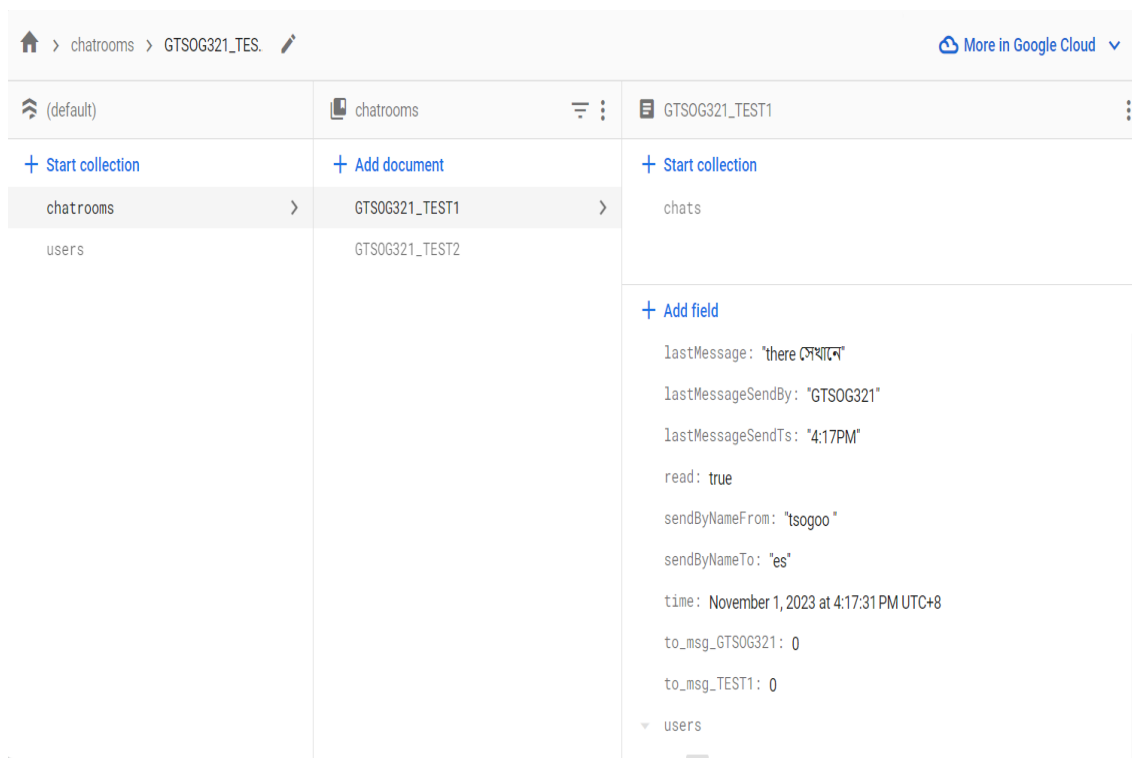
4.2.2 *Firebase* ашиглан өгөгдлийн сангаа бэлдэх

Хамгийн түрүүнд **FirebaseFirestore** өгөгдлийн сангаа үүсгэж, өгөгдлийн сангийн схемийнхээ дагуу өгөгдлөө объект байдлаар бүтэцлэн хадгална.

```
1 class DatabaseMethods {
2     createChatRoom(
3         String chatRoomId, Map<String, dynamic> chatRoomInfoMap) async {
4         final snapshot = await FirebaseFirestore.instance
5             .collection("chatrooms")
6             .doc(chatRoomId)
7             .get();
8         if (snapshot.exists) {
9             return true;
10        } else {
11            return FirebaseFirestore.instance
12                .collection("chatrooms")
13                .doc(chatRoomId)
14                .set(chatRoomInfoMap);
15        }
16    }
17    Future addMessage(String chatRoomId, String messageId,
18        Map<String, dynamic> messageInfoMap) async {
19        return FirebaseFirestore.instance
20            .collection("chatrooms")
21            .doc(chatRoomId)
22            .collection("chats")
23            .doc(messageId)
24            .set(messageInfoMap);
25    }
26 }
```

Код 4.2: Өгөгдлийн сангаа бүтэцлэх мөн объект нэмэх frame

Дээрх функцүүд **хэрэгжсэний** дараа бичсэн моделийн дагуу өгөгдлийн сан дээр **document** бүхий json объектүүд үүссэн.



Зураг 4.2: JSON объектын харагдац

4.2.3 Хэрэглэгчийн ярьсан аудио файлыг орчуулах *End point* гаргах

Back-end хэсгийн кодыг өмнө нь дурдсанчлан шууд Flask фрэймворк дээр тусад нь **repository** үүсгэж апп талаасаа шаардлагатай функцүүдийг хэрэглэж хэрэгтэй өгөгдлүүдийг тодорхой нөхцлүүдээр авч хариуг буцаана.

```

1
2 from flask import Flask, jsonify, render_template, request, redirect,
   session
3 client = Client("https://facebook-seamless-m4t.hf.space/")
4
5 @app.route('/todo', methods=["POST"])
6 def get_todos():
7     try:
8         conv_type=request.form.get('type')
9
10         input_lan=request.form.get('input')
11         output_lan=request.form.get('output')

```

```
12
13     chatroomid=request.form.get('roomId')
14     username=request.form.get('myUsername')
15
16     print('roomId',chatroomid)
17     print('myusername',username)
18
19     print(input_lan)
20     print(output_lan)
21
22
23     # input_lan="English"
24     # output_lan="Halh Mongolian"
25     # translation="S2TT (Speech to Text translation)"
26     if conv_type=="audio":
27         translation=request.form.get('translation')
28         audio_file = request.files['audio']
29         print(translation)
30         if audio_file:
31             #     # Save the audio file to a desired location
32             save_path = os.path.join(os.getcwd(), 'uploads', audio_file
33                                     .filename)
34             audio_file.save(save_path)
35
36             result = translate(save_path, input_lan, output_lan,
37                               translation)
38             #result = translate(save_path)
39             print(result)
40
41         if output_lan == 'Halh Mongolian':
42             # to Mongol start block
43             targeted_languga_text=convertTuple(result)
44             synthesize(targeted_languga_text,chatroomId=chatroomid,
```



```
        username=username)
44         #to Mongol end block
45     else:
46
47         print('to foriegn')
48
49
50         print(result[0])
51         path= result[0]
52
53         # below is speech to speech start
54
55         with wave.open(path, 'r') as wf:
56             # Read audio data
57             audio_data = wf.readframes(-1)
58             print('Audio data read successfully.')
59
60             new_directory = '/home/ubuntu/download/'+chatroomid+'/'
61             +username
62             print(new_directory)
63             print(os.path.exists(new_directory))
64             if not os.path.exists(new_directory):
65                 print('creating direcotry')
66                 os.makedirs(new_directory)
67                 print('created')
68
69             translated_path = os.path.join(os.getcwd(),
70                 new_directory+ '/output.wav')
71             with wave.open(translated_path, 'w') as new_wf:
72                 # Write audio data to the new file
73                 new_wf.setnchannels(wf.getnchannels())
74                 new_wf.setsampwidth(wf.getsampwidth())
75                 new_wf.setframerate(wf.getframerate())
76                 new_wf.writeframes(audio_data)
```

```

75         print('Audio data written to the new file:',
76               translated_path)
77
78         print('4')
79
80     # end s2s
81     return jsonify({'message': get_file_url(chatroomid,username
82                                           )}), 200
83
84     else:
85         return jsonify({'error': 'No audio file provided'}), 400
86
87     else:
88         print('pre translate text')
89         txt=request.form.get('text')
90         result = translate_text(txt, input_lang, output_lang)
91         targeted_languga_text=convertTuple(result)
92         print('translated text',targeted_languga_text)
93         return jsonify({'message': targeted_languga_text}), 200
94     except Exception as e:
95         print(e)
96         return jsonify({'error': str(e)}), 500
97
98     ...

```

Код 4.3: Flask дээрээ end point гаргах

4.2.4 Front-end хэсэг дээр гаргасан End point-оо ашиглаж орчуулсан аудиогоо тоглуулах

Одоо Front-end хэсэг дээрээ орчуулсан аудиогоо тоглуулах шаардлагатай. Үүний тулд dio хүсэлтийг ашиглаж орчуулсан аудио файлынхаа URL хаягийг авч үүнийгээ Firebase-рүү мессеж болгон илгээгээд нөгөө хэрэглэгчдээ уг файлаа татаж аваад тоглуулах юм.

```

1
2 static Future<dynamic> sendAudio(String path, String from, String to,
3   String tranlsation, String chatroomId) async {

```

```

4      final dio = Dio();
5
6      FormData formData = FormData.fromMap({
7          'type': "audio",
8          'audio': await MultipartFile.fromFile(path, filename: 'record.wav'),
9          'input': from,
10         'output': to,
11         'translation': translation,
12         'roomId': chatroomId,
13         'myUsername': _dataController.myUserName
14     });
15
16     final response = await dio.post(
17         url,
18         data: formData,
19         options: Options(headers: {"Content-Type": "multipart/form-data"}),
20     );
21
22     if (response.statusCode == 200) {
23
24         return response.data['message'];
25     } else {
26         log('unsuccessfull req');
27         return 'error';
28     }
29 }

```

Код 4.4: Dio ашиглан хүсэлт илгээж өгөгдлөө татаж авах

```

1
2     ...
3     sendAudioLink(String val) async {
4         String messageId = randomAlphaNumeric(10);
5         DateTime now = DateTime.now();
6         String formattedDate = DateFormat.yMd().format(now);

```

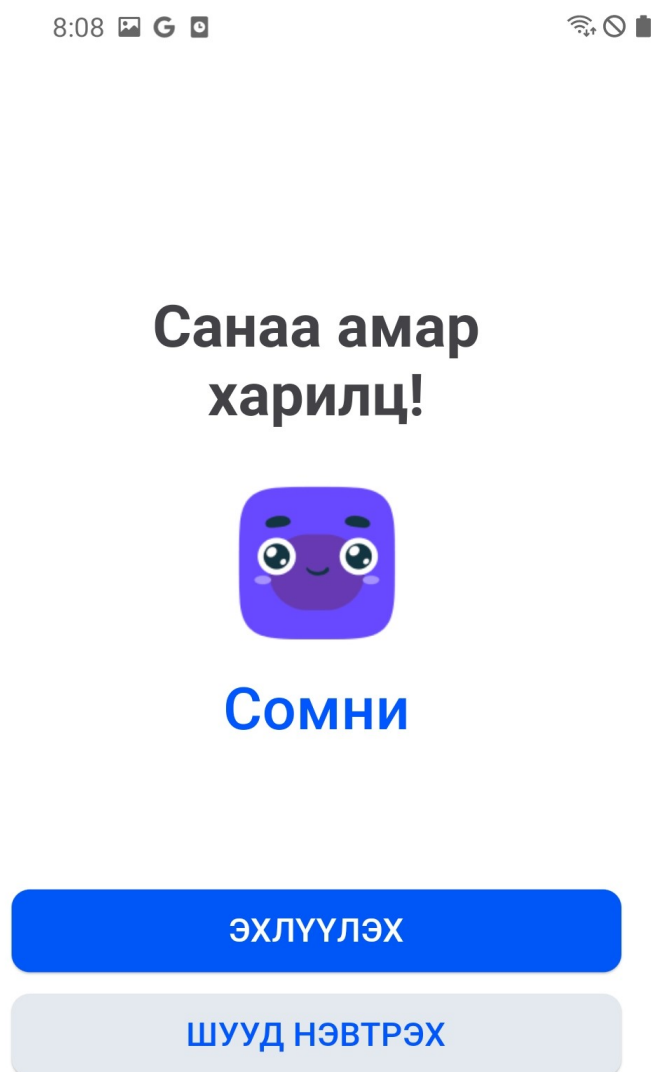
```
7      String hour = DateFormat.Hm().format(now);
8
9      print('video time in vidoe call screen: $formattedDate, $hour');
10     Map<String, dynamic> messageInfoMap = {
11         "id": messageId,
12         "type": "audio",
13         "url": val,
14         "message": "",
15         "sendBy": widget.myUserName,
16         "read": false,
17         "time": FieldValue.serverTimestamp(),
18         "ts": hour + " , " + formattedDate,
19         "missed": false
20     };
21
22     DatabaseMethods().addMessage(widget.channel, messageId, messageInfoMap)
23         ;
24 }
25 ...
```

Код 4.5: FirebaseFirestore-ийн чат өрөөний чатууд руугаа
орчуулсан аудио файлын URL-г илгээх

4.3 Үр дүн

Дээрх үйлдлүүд нь манай төслийг хэрхэн ажиллаж буйг тоймлон харуулсан ба үр дүнд нь хэрэглэгчийн ярьсан яриа болон текст нь түүний сонгосон хэлний дагуу нөгөө хэрэглэгчид орчуулагдах юм.

Мөн цаашид дээрх байдлаар шаардлагын дагуу гарсан интерфэйсүүдийг хийж дуусгах ба платформынхоо зарим зургаас оруулав.





Зураг 4.3: Нүүр хуудас


1:50 M M A •


Ярих хэлээ сонгох


Өөрийн ярьдаг хэлээ сонгоно уу:


 Halh Mongolian


 Bengali


 Catalan


 Czech

 Danish

 Dutch

 English

 Estonian

 German

Үргэлжлүүлэх

(a) Шинэ хэрэглэгчийн хувьд өөрийн
ярьж чаддаг хэлээ сонгох хуудас

8:07 G M

←



Нэр

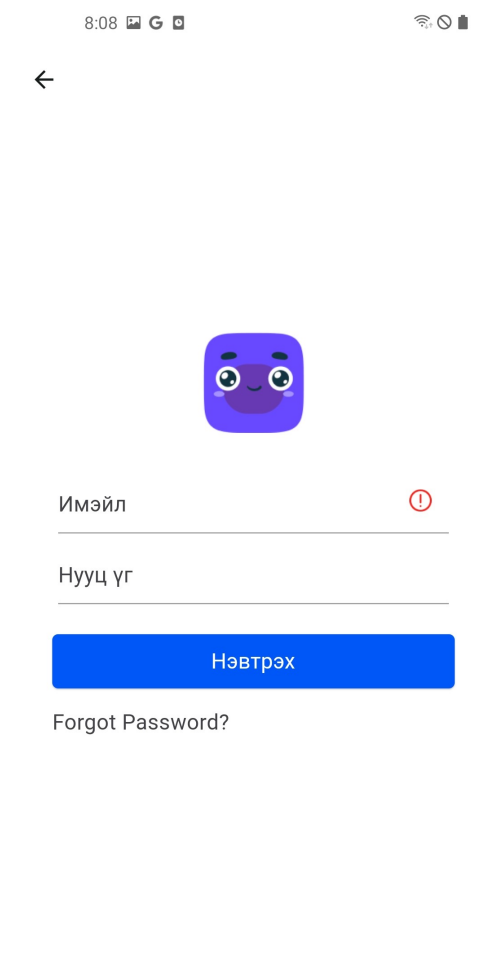
Имэйл

Нууц үг

Нууц үг баталгаажуулах

Бүртгүүлэх

(b) Бүртгүүлэх хуудас



8:08

←

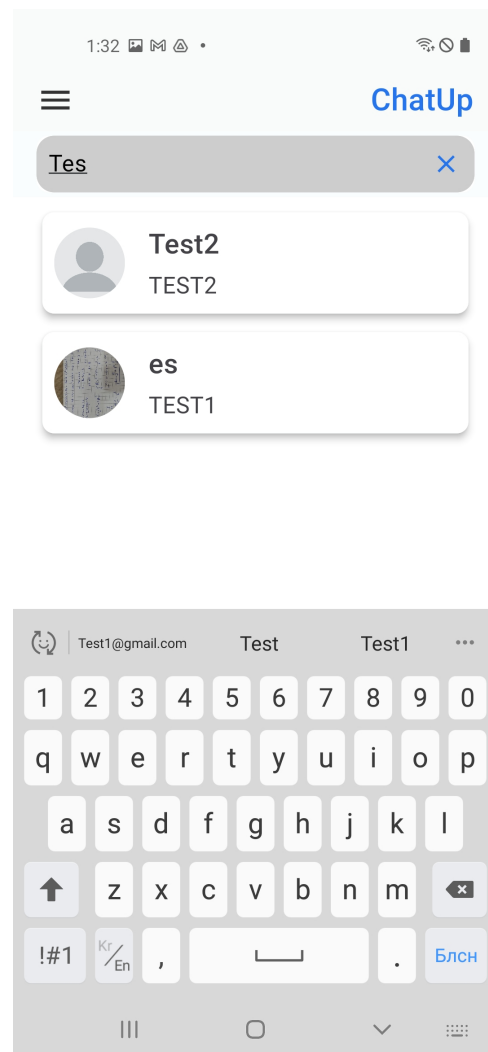
Имэйл

Нууц үг

Нэвтрэх

Forgot Password?

(a) Нэвтрэх хуудас



1:32

ChatUp

Tes

Test2
TEST2

es
TEST1

Test1@gmail.com Test Test1

1 2 3 4 5 6 7 8 9 0

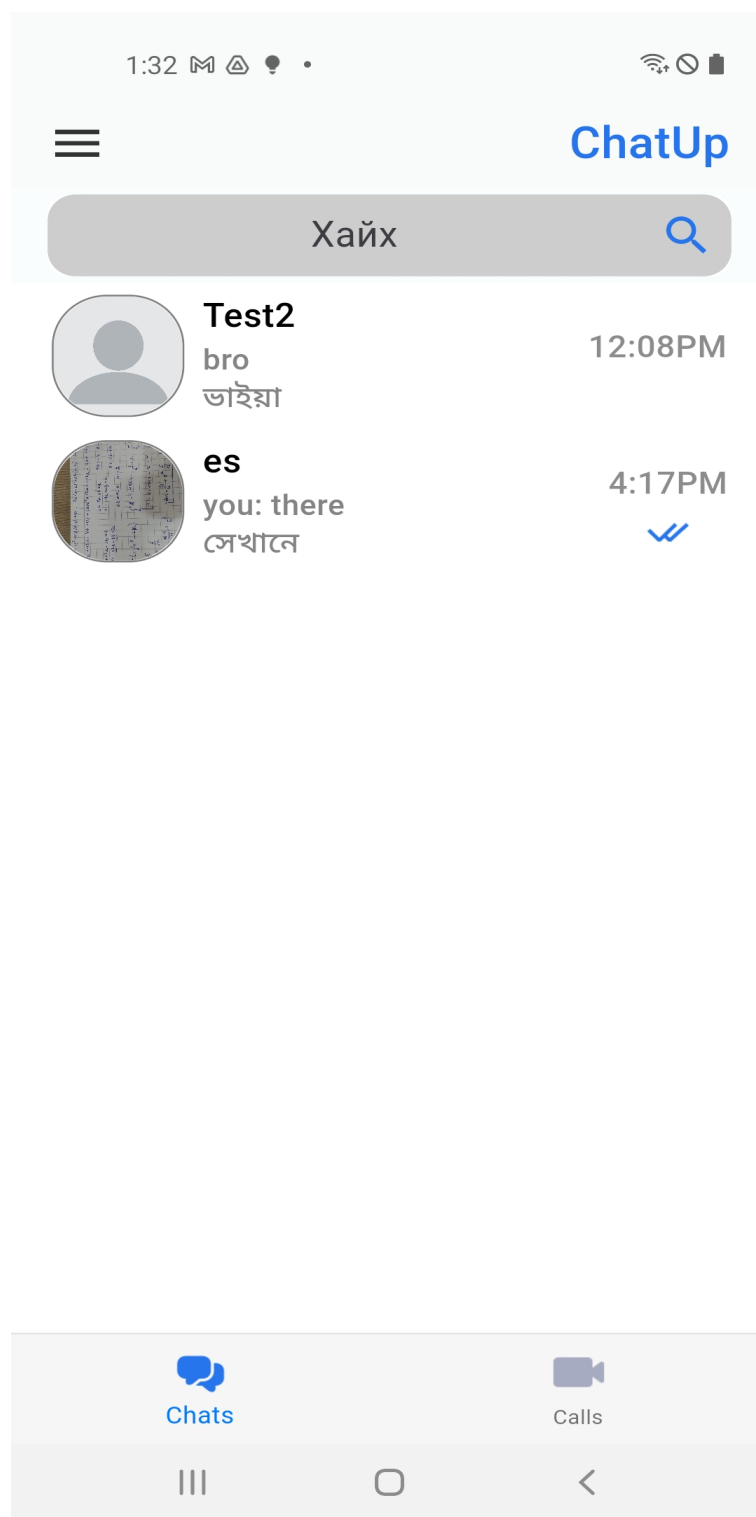
q w e r t y u i o p

a s d f g h j k l

↑ z x c v b n m

!#1 Кг/En , _ . Блсн

(b) Уг аппын хэрэглэгчдийг мэйлээр нь хайх хуудас



Зураг 4.6: Чат хуудас

5. ДҮГНЭЛТ

Энэхүү судалгааны ажлаар орчин үеийн дижитал бүтээгдэхүүний хөгжүүлэлтийн арга барилтай танилцаж хэрэглэгчийн шаардлагыг тодорхойлохоос эхлээд UX судалгаа хийж хэрэглэгч төвтэй интерфэйс дизайн гаргах, шинэлэг технологиудыг ашиглан хөгжүүлэлтийг гүйцэтгэж, өөрийн санаагаа бүрэн ажиллагаатай, хүмүүсийн асуудлыг шийдвэрлэж чадсан сошиал платформ болгож чадсан нь өөрт маань их хэмжээний мэдлэг, туршлага болж үлдсэн гэдэгт итгэлтэй байна.

Мөн цаашлаад уг сошиал платформыг эхний түвшинд LIME аппликейшны видео дуудлагад нэвтрүүлж, дэлхийн монголчууд төдийгүй бусад орны хүмүүс ч гэсэн өөр хоорондоо хэл хамаарахгүй харилцан яриа өрнүүлэх боломжтой систем болгон хөгжүүлэх бололцоотой гэж үзэж байна.

Ашигласан материал

[1] AWS EC2

https://docs.aws.amazon.com/ec2/?nc2=h_q1_doc_ec2

[2] Flutter

<https://pub.dev/>

[3] Flask

<https://flask.palletsprojects.com/en/3.0.x/>

[4] Firebase - Documentation

<https://firebase.google.com/docs/reference/kotlin/com/google/firebase/firestore/DocumentSnapshot>

A. SEAMLESSM4T МОДЕЛ

```
1 client = Client("https://facebook-seamless-m4t.hf.space/")
2
3
4 def translate(url, input_lan, output_lan,translate):
5
6     if output_lan=='Halh Mongolian':
7         result = client.predict(
8             translate, # str in 'Task' Dropdown component
9             "files", # str in 'Audio source' Radio component
10            url, # str (filepath or URL to file) in 'Input speech'
11               Audio component
12            url, # str (filepath or URL to file)in 'Input speech'
13               # Audio component
14            "hi!", # str in 'Input text' Textbox component
15            input_lan, # str in 'Source language' Dropdown
16               component
17            output_lan, # str in 'Target language' Dropdown
18               component
19            api_name="/run"
20        )
21
22     return result
23
24 else:
25
26     result = client.predict(
27         translate, # str in 'Task' Dropdown component
28         "files", # str in 'Audio source' Radio component
29         url, # str (filepath or URL to file) in 'Input
30            speech' Audio component
31         url, # str (filepath or URL to file)in 'Input
32            speech'# Audio component
33         "hi!", # str in 'Input text' Textbox component
34         input_lan, # str in 'Source language' Dropdown
35            component
36         output_lan, # str in 'Target language' Dropdown
37            component
38         api_name="/run"
39     )
40
41     return result
```

B. CHIMEGE -Г АРІ-ИАР АШИГЛАХ

```
1 def synthesize(text, chatroomId, username):
2     url = "https://api.chimege.com/v1.2/synthesize"
3     headers = {
4         'Content-Type': 'plain/text',
5         'Token': '7769
6             d0fe9a57fda0588cae44dff6f469ad1ea464a003a0f004034c3443f9fe40',
7     }
8     r = requests.post(
9         url, data=text.encode('utf-8'), headers=headers)
10
11     with open("/home/ubuntu/download/"+chatroomId+"/"+username+"/output.wav", 'wb') as out:
12         out.write(r.content)
```

C. FLUTTER ДЭЭР AWS-ДЭЭРХ СЕРВЕР ЛҮҮ ХАНДАХ

```
1 static Future<String> sendText(String text, String from, String to) async
2 {
3   final dio = Dio();
4   final url = 'http://51.20.44.63:5000/todo';
5   FormData formData = FormData.fromMap({
6     'type': "text",
7     'text': text,
8     'input': from,
9     'output': to,
10  });
11
12  print('pre response for text $text |$from| {$to}');
13
14  final response = await dio.post(
15    url,
16    data: formData,
17    options: Options(headers: {"Content-Type": "multipart/form-data"}),
18  );
19  log('response: ${response}');
20
21  if (response.statusCode == 200) {
22    print('translated: ${response.data['message']}');
23
24    return response.data['message'];
25  } else {
26    log('unsuccessfull req');
27    return 'error';
28  }
29 }
```

D. ХЭРЭГЛЭГЧИЙН AGORA TOKEN ҮҮС-ГЭХ ENDPOINT

```
1 @app.route('/generate_agora_token/<channelName>', methods=['POST'])
2 def generate_token(channelName):
3
4
5     # Your Agora App ID and App Certificate
6     app_id = "d565b44b98164c39b2b1855292b22dd2"
7     app_certificate = "caf2f127d2a64a5d92afaf7aee8b3609"
8     # User ID (Optional)
9     user_id = request.form.get('uid')
10
11     expireTimeInSeconds = 3600
12     currentTimeStamp = int(time.time())
13     privilegeExpiredTs = currentTimeStamp + expireTimeInSeconds
14
15     token = RtcTokenBuilder.buildTokenWithUid(
16         app_id, app_certificate, channelName, user_id, 1,
17         privilegeExpiredTs)
18     return jsonify({'token': token})
```

E. ЧАТЫН МЭДЭГДЭЛ ИЛГЭЭХ ENDPOINT

```
1 @app.route("/sendChat", methods=['GET'])
2 def send_Notification():
3
4     fcm=request.form.get('fcm')
5     name=request.form.get('name')
6     content=request.form.get('content')
7     print('send notification:'+fcm+' name:'+name+' content:'+content)
8
9
10
11
12 # if not app_init:
13
14     #app_init=True
15
16     # Create a message with the notification payload
17     message = messaging.Message(
18         notification=messaging.Notification(
19             title=name,
20             body=content,
21         ),
22
23         token=fcm, # Replace with the FCM token of the target device
24     )
25
26     # Send the message
27     response = messaging.send(message)
28     print("Successfully sent message:", response)
29     return response
```

Ү. ЧАТ ӨРӨӨ БҮРИЙН ХУВЬД ШИНЭ ЧАТЫГ СОНСОХ EVENT

```
1 void listenForNewMessages(String channel, String username,  
2 List<String> user_native_lans, BuildContext context) {  
3   final CollectionReference messagesCollection =  
4     FirebaseFirestore.instance.collection('chatrooms/${channel}/chats')  
5     ;  
6   NewMessages[channel] =  
7     messagesCollection.snapshots().listen((QuerySnapshot snapshot) {  
8       snapshot.docChanges.forEach((change) async {  
9         final messageData = change.doc.data() as Map<String, dynamic>;  
10  
11         if (!processedMessageIds.contains(messageData['id'])) {  
12           bool exited = exitedForEachChannel_Voice[username] ?? true;  
13           print(  
14             'new message: ${messageData}, widget username: ${username},  
15               exited: ${exited}');  
16  
17           if (messageData["type"] == 'request' &&  
18             messageData["sendBy"] != myusername &&  
19             messageData["rejected"] as bool == false &&  
20             messageData["accept"] as bool == false &&  
21             exited) {  
22             await SomniAlerts.alertBoxVertical(  
23               onClose: () async {  
24                 Get.back();  
25                 try {  
26                   final chatPairRef = FirebaseFirestore.instance  
27                     .collection('chatrooms')  
28                     .doc(channel)  
29                     .collection('chats')  
30                     .doc(messageData["id"]);  
31                   await chatPairRef.update({  
32                     'rejected': true,  
33                   });  
34                 } catch (e) {  
35                   print('Error updating chat pair: $e');  
36                 }  
37               },  
38               titleWidget: Container(  
39                 width: double.infinity,  
40                 padding: EdgeInsets.symmetric(horizontal: 20),  
41                 child: Text(  
42                   maxLines: 2,  
43                   overflow: TextOverflow.ellipsis,  
44                   softWrap: true,  
45                   '',  
46                   textAlign: TextAlign.center,  
47                   style: TextStyle(  
48                     color: Colors.black,  
49                     fontSize: 24,  
50                     fontWeight: FontWeight.w500),  
51                 ),  
52               textWidget: Padding(  
53                 padding: const EdgeInsets.symmetric(horizontal: 20),  
54                 child: RichText(  
55                   textScaleFactor:
```



```

56         MediaQuery.of(Get.context!).textScaleFactor,
57         textAlign: TextAlign.center,
58         text: TextSpan(
59             text: 'Видео дуудлага',
60             style: TextStyle(
61                 color: Colors.black,
62                 fontSize: 13,
63                 fontWeight: FontWeight.w400),
64             children: <TextSpan>[
65                 TextSpan(
66                     text: 'Танд',
67                     style: TextStyle(fontWeight: FontWeight.bold)),
68                 TextSpan(text: '$username хэрэглэгчээс'),
69                 TextSpan(text: 'видео дуудлагаиржбайна .'),
70             ],
71         )),
72     ),
73     button1: () async {
74         int intValue = Random().nextInt(10000);
75         String token = await Data.generate_token(channel, intValue)
76         ;
77         String key = channel + myusername;
78         Customer? user = usersBox.get(key);
79         String from = '', to = '';
80         if (user != null) {
81             from = user.trans_from_voice;
82             to = user.trans_to_voice;
83         } else {
84             usersBox.put(
85                 channel,
86                 Customer(
87                     id: '1',
88                     trans_from_voice: 'Halh Mongolian',
89                     trans_to_voice: user_native_lans[0],
90                     trans_from_msg: 'Halh Mongolian',
91                     trans_to_msg: user_native_lans[0],
92                 ));
93             from = 'Halh Mongolian';
94             to = user_native_lans[0];
95         }
96         Get.to(Video_call_screen(
97             channel, myusername, username, from, to, token,
98             intValue));
99         try {
100             final chatPairRef = FirebaseFirestore.instance
101                 .collection('chatrooms')
102                 .doc(channel)
103                 .collection('chats')
104                 .doc(messageData["id"]);
105             await chatPairRef.update({
106                 'accept': true,
107             });
108         } catch (e) {
109             print('Error updating chat pair: $e');
110         }
111         Navigator.of(context).pop();
112     },
113     button2: () async {
114         try {
115             final chatPairRef = FirebaseFirestore.instance
116                 .collection('chatrooms')
117                 .doc(channel)
118                 .collection('chats')

```

```

117         .doc(messageData["id"]);
118         await chatPairRef.update({
119             'rejected': true,
120         });
121     } catch (e) {
122         print('Error updating chat pair: $e');
123     }
124     Get.back();
125 },
126 imgAsset: 'alert/alert_reminder',
127 button1Text: 'Дуудлага авах□□ ',
128 button2Text: 'Дууллага салгах□□ ',
129 );
130 }
131 if (messageData["type"] == "audio" && exited) {
132     updateChatReadState(messageData["id"], false, true, channel);
133 } else if (messageData["type"] == "audio" &&
134     messageData["sendBy"] == username &&
135     messageData["missed"] == false &&
136     messageData["read"] == false) {
137     downloadAndPlayAudio(
138         messageData["url"], messageData["id"], channel);
139     }
140     processedMessageIds.add(messageData['id']);
141     // Process and display the new message as needed
142 }
143 });
144 });
145 }

```