

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

Цэрэннадмид Намсрайжамц

**Mevent - Эвэнт зохион байгуулах апликашн
(Mevent - Event management application)**

Програм хангамж (D061302)
Бакалаврын судалгааны ажил

Улаанбаатар

2022 он

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

**Mevent - Эвэnt зохион байгуулах архивийн
(Mevent - Event management application)**

Програм хангамж (D061302)
Бакалаврын судалгааны ажил

Удирдагч: _____ Mr. Н.Даваасүрэн

Гүйцэтгэсэн: _____ Ц.Намсрайжамц (18B1NUM1671)

Улаанбаатар

2022 он

Зохиогчийн баталгаа

Миний бие Цэрэннадмид Намсрайжамц ”Mevent - Эвэnt зохион байгуулах аппликеийшн” сэдэвтэй судалгааны ажлыг гүйцэтгэсэн болохыг зарлаж дараах зүйлсийг баталж байна:

- Ажил нь бүхэлдээ эсвэл ихэнхдээ Монгол Улсын Их Сургуулийн зэрэг горилооор дэвшүүлсэн болно.
- Энэ ажлын аль нэг хэсгийг эсвэл бүхлээр нь ямар нэг их, дээд сургуулийн зэрэг горилооор 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

УДИРТГАЛ	1
1. СИСТЕМИЙН СУДАЛГАА	3
1.1 Ижил төстэй системийн судалгаа	3
1.2 Технологийн судалгаа	6
1.3 Хэрэгцээ шаардлага	8
2. СИСТЕМИЙН ШИНЖИЛГЭЭ	10
2.1 Системийн ерөнхий тодорхойлолт	10
2.2 Системийн шаардлага	11
2.3 Системийн ерөнхий зохиомж	14
2.4 Өгөгдлийн урсгалын шинжилгээ	14
2.5 Объект хандлагат шинжилгээ	16
2.6 Өгөгдлийн сангийн зохиомж	18
3. СИСТЕМИЙН ЗОХИОМЖ	23
3.1 Интерфэйсийн зохиомж	23
4. ХЭРЭГЖҮҮЛЭЛТ	32
4.1 Системийн хөгжүүлэлт	32
4.2 Бүртгүүлэх болон нэвтрэх хэсэг	33
4.3 Эвэnt үүсгэх	36
ДҮГНЭЛТ	42
НОМ ЗҮЙ	42
ХАВСРАЛТ	43
A. КОДЫН ХЭРЭГЖҮҮЛЭЛТ	44
A.1 Back-side хэрэгжүүлэлтийн код	44
A.2 Client-side хэрэгжүүлэлтийн код	53

ЗУРГИЙН ЖАГСААЛТ

1.1	Meetup эвэнтэд бүртгүүлэх хэсэг	4
1.2	Meetup гар утасны апликеийшн төрөл сонгох	5
2.1	Ерөнхий диаграм	14
2.2	Use case диаграм	16
2.3	Класс диаграм	17
2.4	Хэрэглэгч - өгөгдлийн тайлбар	19
2.5	Зохион байгуулагч - өгөгдлийн тайлбар	19
2.6	Захиалга - өгөгдлийн тайлбар	20
2.7	Эвэнтийн төрөл - өгөгдлийн тайлбар	20
2.8	Эвэnt - өгөгдлийн тайлбар	21
2.9	Дагагчид - өгөгдлийн тайлбар	21
2.10	Таалагдсан эвэнтүүд - өгөгдлийн тайлбар	22
2.11	Байршил - өгөгдлийн тайлбар	22
2.12	Тасалбар - өгөгдлийн тайлбар	22
3.1	Апликеийшн ажиллаж эхлэхэд харагдах хэсэг болон нүүр хуудас	24
3.2	Эвэnt болон зохион байгуулагч	25
3.3	Нэвтрэх болон бүртгүүлэх хэсэг	26
3.4	Хэрэглэгчийн хэсэг	27
3.5	Тасалбар болон таалагдсан	28
3.6	Тасалбар болон qf код	29
3.7	Хайлтын хэсэг	30
3.8	Дагасан зохион байгуулагчдыг харуулах хэсэг	31
4.1	Хөгжүүлсэн API жагсаалт	33
4.2	Бүртгүүлэх	34
4.3	Бүртгүүлэх - Өгөгдлийн санд	35

4.4	Нэвтрэх	36
4.5	Нэвтрэх - API буцаалт	36
4.6	Эвэнт үүсгэх	37
4.7	Эвэнт үүсгэх - Өгөгдлийн санд	38
4.8	Тасалбар худалдан авах	39
4.9	Тасалбар худалдан авах - Өгөгдлийн санд	39
4.10	Хайлт хийх	40
4.11	Таалагдсан болон дагасан	41

Кодын жагсаалт

A.1	Хэрэглэгч бүртгүүлэх нэвтрэх	44
A.2	Хэрэглэгч бүртгүүлэх болон нэвтрэх үед нууцлалыг хэрэгжүүлэх	45
A.3	Зураг оруулахтай холбоотой API	47
A.4	Эвэнттэй холбоотой API	49
A.5	Эвэntийн apи- аар дамжуулах авах өгөгдлийг бэлдэж өгөх	50
A.6	Аппликейшны нүүр хэсгийг харуулах	53

УДИРТГАЛ

СЭДВИЙГ СОНГОХ БОЛСОН ҮНДЭСЛЭЛ

Баяр ёслол, тэмцээн уралдаан, сургалт семинар, урлаг спорт, эрүүл мэнд , боловсрол гэх мэт олон сэдвийн хүрээнд ямар нэгэн үйл ажиллагаанууд өдөр болгон хaa нэгтэй зохион байгуулагдаж байдаг. Тэдгээр үйл ажиллагааны талаарх мэдээллүүд болон зарлалууд нь ихэвчлэн танилын хүрээнд, Facebook - ийн бүлэгт эсвэл хэн нэгэн алдартай хүний Instagram story -гоор хүмүүст хүрдэг.

Нэг хүний үнэхээр оролцохыг хүсэж байсан үйл ажиллагаа нь тэр хүний танилын хүрээгээр дамжаагүй, facebook-ийн бүлэгт нь зарлагдаагүйгээс үүдэн тухайн хүн тэр үйл ажиллагааны талаарх мэдээллийг дууссаных нь дараа мэдсэн.

Миний энэ сэдвийг сонгох болсон нь энэ хүнд тохиолдсон шиг асуудлыг шийдэх мөн ямар нэгэн үйл ажиллагаа зохион байгуулах үйл явцыг илүү боловсронгуй болгох сэдэл дээр үндэслэгдсэн юм.

ЗОРИЛГО

Мобайл хэрэглэгчид сонирхсон аливаа үйл ажиллагааны тухай мэдээллийг түргэн шуурхай авах, тэдгээрт оролцох боломжийг хялбаршуулах, үйл ажиллагааны төлбөр тооцоогоо мобайлаар хийх, үйл ажиллагаа зохион байгуулагчид нь мэдээллээ олон хүнд хүргэх, тасалбар борлуулалтыг хялбар хийх шийдэл бүхий мобайл аппликеийн хөгжүүлэх зорилготой.

ЗОРИЛТ

Энэхүү зорилгынхoo хүрээнд дараах зорилтуудыг тавьсан

1. Үйл ажиллагаа зохион байгуулах процесийн талаар судлах, ижил төстэй системүүдийг судлах
2. Системийн хэрэгцээ, шаардлагыг тодорхойлох
3. Системийг бүтээх технологийн судалгаа хийх

4. Системийн шинжилгээ хийх, зохиомжийг гаргах
5. Тодорхойлсон шаардлага, зохиомжийн дагуу апликеийшн хөгжүүлэх

АШИГЛАСАН ТЕХНОЛОГИ

Апликеийшн хөгжүүлэхийн тулд дараах технологиудыг ашигласан.

- Жаваскрипт програмчлалын хэл (JavaScript)
- React.js (сан)
- React Native (фраймворк)
- Java Spring Boot (фраймворк)
- Postman (API сервис ажиллуулах, турших хэрэгсэл)
- AdobeXD (Хэрэглэгчийн интерфэйсийн зохиомж гаргах хэрэгсэл)
- PostgreSQL (Өгөгдлийн сан)

АЖЛЫН ҮР ДҮН, АЧ ХОЛБОГДОЛ

Үйл ажиллагаа зохион байгуулагч болон үйл ажиллагаанд оролцогчдын хэрэгцээ шаардлагад нийцэхүйц тэдгээр хоёр талыг хооронд холбосон апликеийшн бий болсон байна. Зохион байгуулаг - чийн зүгээс өөрийн зохиох гэж буй үйл ажиллагааны дэлгэрэнгүй мэдээллийг оруулж үйл ажиллагааг нийтлэх , оролцогчийн хувьд нийтлэгдсэн үйл ажиллагаанаас сонгон төлбөрөө төлөн оролцох боломжтой.

1. СИСТЕМИЙН СУДАЛГАА

Өөрийн бүтээх гэж буй системтэй ижил төрлийн системийн ажиллагаа болон бүтэц мөн систем бүтээхдээ ямар төрлийн технологийг хэрхэн ашиглах талаарх судалгаануудыг хийж гүйцэтгэсэн.

1.1 Ижил төстэй системийн судалгаа

Олон улсад үйл ажиллагаа зохион байгуулах энэ төрлийн систем багагүй байдаг ба ашигласан технологи болон боломжууд, үйл ажиллагааны процессуудыг судлан үзсэн. Судалгааны явцад хэд хэдэн төстэй **Meetup**, **Eventbrite**, **Locals** гэх зэрэг системүүд бусдаасаа илүү олонд танигдсан байсан. Эдгээрийн дундаас **Meetup** гэх системийн голлон судалсан.

1.1.1 Төстэй системүүдийн нийтлэг үндсэн үйл ажиллагаа

Ихэнх эвэnt зохион байгуулах системийн ерөнхий үйл ажиллагаа нь системийг хэрэглэж буй хэрэглэгчдийн газар зүйн байршил болон сонирхсон төрлийнх нь дагуу эвэнтүүдийг хэрэглэгчдэд санал болгон хэрэглэгчдийг идэвхжүүлэх, мөн хэрэглэгчид өөрийн таалагдсан эвэнтээ хадгалах бусадтай хуваалцах зэрэг боломжуудыг олгосон бүтэцтэй байна.

Ижил төстэй системүүдийн ажиллах зарчим, онцлогийг судлан үзсэний үндсэн дээр дараах нийтлэг хэрэгжүүлж болохуйц боломжуудыг жагсаалтыг гаргав.

- Хэрэглэгчийн сонирхдог төрлийн дагуу эвэнтүүдийг харуулах
- Хэрэглэгчдийн сонирхлынх нь дагуу бүлэг үүсгэх
- Хайлтын хэсэг нь олон төрлөөр хайх боломжтой
- Сонирхож буй эвэнтийн мэдээллийг найзууттайгаа хуваалцах боломж
- Эвэнт зохион байгуулагчийн тасалбарын борлуулалт болон санхүүгийн анализыг харуулах

- Тасалбар шалгахдаа QR код ашиглах

1.1.2 Meetup

Meetup нь ижил төстэй сонирхолтой хүмүүст зориулсан эвэnt зохион байгуулдаг веб болон гар утасны аппликашн. **Meetup**-ийг 2002 онд **Scott Heiferman** гэх хүн дөрвөн үүсгэн байгуулагчдын хамтаар үүсгэн байгуулсан. 2017 оны тооцооллоор **35'734'000** гаруй хэрэглэгчтэй байсан бөгөөд ихэвчлэн барууны орнуудын хүмүүс хэрэглэдэг байна. **Meetup**-ийн гол боломж нь хэрэглэгч өөрийн сонирхлын дагуу бүлэг үүсгэх эсвэл бүлэгт элсэх замаар эвэнтүүдэд оролцож эсвэл зохион байгуулдаг .

Meetup харагдах байдал

The screenshot shows a Meetup event page for "Tokyo Alternative Music and Musicians Discord". At the top, there's a navigation bar with the Meetup logo, a search bar, and a location dropdown set to "Ulaanbaatar, MN". On the right, there are buttons for "Start a new group - 30% off!", messaging, and profile icons.

The main content area displays the event details:

- Sunday, May 1, 2022**
- Tokyo Alternative Music and Musicians Discord**
- Hosted By**: Aaron (with a small profile picture)

Below the title, there's a grid of 15 small images representing various music genres or styles.

To the right of the grid, there's a sidebar with the following information:

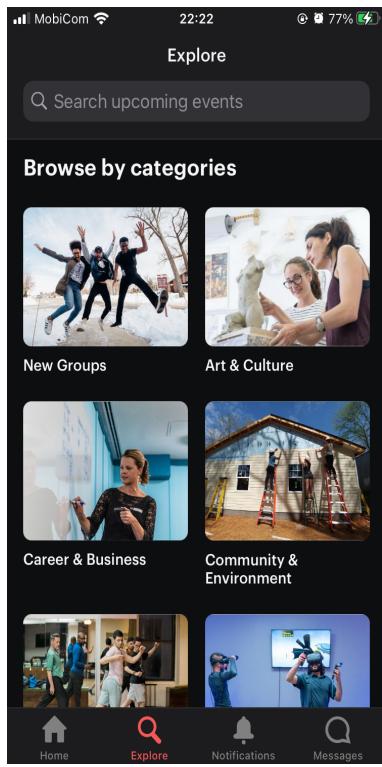
- Tokyo Alternative Music and Musicians Discord** (Public group)
- Sunday, May 1, 2022**: 12:00 PM to 1:00 PM ULAT. Every week on Sunday. Add to calendar.
- Online event**: Link visible for attendees

At the bottom of the sidebar, there's a "Report this event" link.

Below the sidebar, there's a "Details" section with the text: "Just hit the link to join the discord and start meeting other musicians and chatting".

At the very bottom, there's a footer with the date "SUN, MAY 1 - 12:00 PM ULAT", the group name "Tokyo Alternative Music and Musicians Discord", a "FREE" status, "19 spots left", a star icon, a "Share" button, and a red "Attend online" button.

Зураг 1.1: Meetup эвэнтэд бүртгүүлэх хэсэг



Зураг 1.2: Meetup гар утасны апликеийшн төрөл сонгох

Meetup боломжууд

Meetup нь илүү хүмүүсийн бүлэг үүсгэхэд чиглэсэн бөгөөд тэр талын боломжуудыг бүтээгдэх-үүндээ илүү тусгаж өгсөн байна. Meetup нь дараах үндсэн боломжуудтай:

- Хэрэглэгч өөртэйгөө сонирхол нэгтэй хүмүүсийн бүлэгт элсэх
- Хэрэглэгч тодорхой төлбөр төлөн бүлэг үүсгэх
- Хэрэглэгч үүсгэсэн бүлэгтэй бусад хэрэглэгчдийг урих
- Хэрэглэгч үүсгэсэн бүлэгтэй эвэнт үүсгэх
- Эвэнтийн дэлгэрэнгүй мэдээлэл дээр сэтгэгдэл бичих болон эвэндээт үнэлгээ өгч болох
- Газарзүйн байрлал болон түлхүүр үгээр эвэнтийг хайх
- Бүлгийн гишүүд хоорондоо чат бичиж харилцах

зэрэг боломжуудыг бүтээгдэхүүндээ шингээсэн байна.

1.2 Технологийн судалгаа

Энэхүү аппыг хөгжүүлэхдээ ашигласан технологи болон бусад хэрэгслүүд, тэдгээрийн талаарх судалгааг танилцуулж байна.

1. Жаваскрипт(Javascript)

Судалгааны ажлын хөгжүүлэлтийн back-end болон хэрэглэгчийн интерфэйс хөгжүүлэх хэрэгжүүлэлтийн аль аль нь жаваскрипт хэл дээр хийгдсэн. Жаваскриптыг хангалттай хүнд бодлогыг шийдэж чадах бүрэн хэмжээний програмчлалын хэл гэж үздэг. Товчхондоо вэб хуудсанд динамик байдлыг бий болгох олон технологийн нэг нь жаваскриптийн. Жаваскриптийг ашигласнаар статик HTML хуудсууд өндөр чанартай хөдөлгөөнт бүрдүүлэлт, өгөгдлийн оролт гаралтын интерактив диалог, өгөгдлийг серверт дамжуулахаас өмнө урьдчилан шалгах боломжуудаар хангагдана.

Жаваскриптийн онцлог талууд:

- Объектод тулгуурласан скрипт хэл дахин тодорхойлсон объектуудыг агуулдаг.
- Вэб хөтөч дээрх хэрэглэгчийн удирдлагыг илүү нэмэгдүүлж өгдөг.
- Огноо, цагийг хялбар зохицуулж чаддаг.
- Хэрэглэгчийн вэб хөтөч болон үйлдлийн системийг илрүүлэх боломжтой.
- Жаваскриптийн бүх төлөв цэг таслалаар төгссөн байх ёстой.

2. Реакт (React.js)

Реакт нь Хэрэглэгчийн Интерфэйс (User Interface) бүтээх зориулалттай Жавастриптын сан (JAVASCRIPT LIBRARY) бөгөөд 2012 онд Фэйсбүүк компаний программист Jordan Walker бүтээжээ. React ашиглан бүх төрлийн апп хөгжүүлэх боломжтой. Жишээлбэл React-DOM санг ашиглан вэб апп хийх боломжтой бол React Native санг ашиглан мобайл

апп хөгжүүлэх боломжтой. React дээр хөгжүүлэлт хийхдээ аплайкейшнийг компонентуудад хувааж бүтээдэг. Компонентийн харагдах хэсгийг JSX хэмээх Жаваскриптын өргөтгөлийг ашиглан бичдэг. Зураг 1.3-аас харвал Reactыг аплайкейшны бүх хэсэгтээ (component) эсвэл зарим хэсэгтээ(component) ч ашиглах тохиолдол бий. Мөн Нэг хуудаст аплайкейшн (Single Page Application) болон Нэг хуудаст аплайкейшн (Multi Page Application) аль ч архитектур реактыг ашиглах боломжтой юм.

3. Java Spring boot

Spring boot нь микро сервис бүтээхэд ашигладаг нээлттэй эх сурвалж дээр суурилсан Java фрэймворк юм. Энэ нь Pivotal баг боловсруулж, зогсож, үйлдвэрлэлийн бэлэн хаврын патрограмыг бий болгоход ашиглагддаг. Spring boot нь Java хөгжүүлэгчдэд сайн платформыг бий болгоход туслах ба ”Just run” гэх уриа үгээр анх танигдаж байсан. Энэ нь ашиглахад хамгийн хялбар, хамгийн хурдан гэсэн санааг илэрхийлж байгаа юм. Та Spring тохиргооны хэрэгцээг хангах шаардлагагүй ба хамгийн бага тохиргоогоор эхлүүлж болно.

4. React native

Фэйсбуүк компани дотооддоо ашиглаж байсан технологио 2013 онд танилцуулсан нь програмчлалын Javascript хэлийг ашиглаж хийсэн Front-end library болох React native технологи юм. Declarative UI хөгжүүлэлтийн аргыг хамгийн анх дэлгэрүүлж, өргөн хэрэглээнд нэвтрүүлж чадсан тул Declarative UI-н гол төлөөлөгч гэж явдаг. Уг технологийг ашиглахын тулд үндсэн хэдэн ойлголтууд авах хэрэгтэй. Үнд component ба түүний lifecycle, javascript-н өргөжүүлсэн хувилбар болох jsx, мөн хамгийн чухал зүйл болох Virtual DOM нар багтана. Declarative UI гэдэг нь хэрэглэгчийн интерфэйсийн кодыг бичихдээ юу зурагдах буюу render хийх үеийн интерфэйсийг бүгдийг урьдчилан тодорхойлдог. Imperative програмчлалаас ялгаатай нь хязгаартай нөхцөлд яг юу хийхийг хатуугаар зааж өгөхгүйгээр тухайн state-с хамааруулж хэрэглэгчийн хүссэн зүйлийг гаргаж өгөх боломжтой. React нь component-based буюу DOM дээр хэвлэж байгаа

бүх зүйлс component байна гэсэн дүрмийг баримталдаг. Component үүсгэж бичихийн давуу тал нь нэг бичсэн кодоо олон дахин бичигдэхээс зайлсхийж, дахин ашиглах боломжийг олгодог. Тус бур өөрсдийн гэсэн дотоод төлөвтэй мөн гаднаас утга хүлээн авах чадвартай. Үүнийг бид Props гэж нэрлэдэг. Мөн component нь stateless, stateful гэж хоёр хуваагддаг ба stateful component нь өөрийн гэсэн төлөвтэй, түүнийгээ удирддаг, class болон hook ашигласан функцууд байна. React-н давуу тал нь state эсвэл props-н өөрчлөлтийг үргэлж хянаж байдаг тул өөрчлөлт орж ирэхэд бүтэн хуудсыг зурах бус зөвхөн тухайн өөрчлөгдсөн component-г л дахин зурдаг. Ингэснээр энгийн вэбүүдээс илүү хурдтай ажилладаг. JSX нь Javascript Extended гэсэн үгний товчлол бөгөөд энгийнээр javascript дотор HTML- н тагуудыг бичиж өгөх мөн кодыг илүү богино болгож хүссэн үр дүндээ хүрэх боломжийг олгодог. Үүний цаана Babel гэсэн transcompiler-г ашиглаж дундын хөрвүүлэлтийг хийдэг ба хэдийгээр HTML таг бичиж байгаа харагддаг ч код дунд цэвэр HTML-г огтоос бичиж өгдөггүй гэсэн үг юм.

5. Adobe XD

Adobe XD нь вектор дээр суурилсан веб болон мобайл applicейшны загвар, хар зургийг гаргахад зориулсан Adobe Inc ээс гаргасан програм хангамж юм. Үүний тусламжтайгаар бодит applicейшн ажиллаж буй мэт ажлын урсгалыг төлөвлөж, хэрэглэгчийн интерфэйсийг бүрэн дүрслэх боломжтой юм.

6. Postman

HTTP протокол ашиглан/get,post,put,delete/гэх мэт аргуудаар(method)хүсэлт илгээж өгөгдөл хүлээн авдаг. Өөрөөр хэлбэл REST API сервисийг төрөл бүрээр дуудаж үр дүнг харах боломжтой хэрэгсэл юм.

1.3 Хэрэгцээ шаардлага

Одоо үед хүмүүс маш олон төрлийн зүйлсийг сонирхдог болсон. Тэрийгээ дагаад өөртэйгөө ижил сонирхолтой хүмүүстэй хамт байх тэдэнтэй илүү ойр дотно болохыг хүсдэг. Тэдний

хүсэлд нийцсэн, тэдний дуртай зүйл бол сонирхол нэгтэй хүмүүстэйгээ хамт байх боломжийг олгодог эвэнтүүд юм. Харин тухайн эвэнтүүдийг хаанаас олж мэдэх вэ? эсвэл өөрөө хэрхэн зохиож түүндээ олон хүмүүсийг татан оролцуулах вэ ? Эндээс миний бүтээхээр сонгосон системийн хэрэгцээ гарч ирж байгаа юм.

Хүмүүс эвэнтийн талаарх мэдээллийг ихэвчлэн сошиал платформуудын (**facebook, instagram, twitter** г.м) эсвэл худалдааны платформын (**shoppy.mn**) тусlamжтайгаар олж мэддэг. Ингэхдээ шүүлтүүр хийн хайж өөрийн сонирхлын дагуу эвэнтийг олж авахад төвөгтэй бөгөөд зарим тохиолдолд олохгүй байх нь элбэг. Хайж хайж олсны эцэст тухайн эвэнтэд бүртгүүлэх болон тасалбарыг худалдан авахын тулд тухайн эвэнтийн зохион байгуулагч талтай утас ,чат зэргээр харилцах хэрэгтэй болдог. Энэ үйл явцуудыг нэгтгэн төвлөрүүлж илүү сайжруулж хэрэглэгчдэд сэтгэл ханамж төрүүлэхүйц систем бүтээх шаардлагатай ба тухайн систем нь дараах давуу талуудыг бий болгоно.Үүнд:

- Хэрэглэгч зохион байгуулагч талтай заавал харилцах шаардлагагүй болж тасалбар худалдан авах, эвэнтийн талаарх дэлгэрэнгүй мэдээлэл болон эвэnt зохион байгуулагчийн өмнө зохиож байсан эвэнтүүдийн талаарх мэдээллийг нэг доорос авах боломжтой болно.
- Ижил сонирхолтой хүмүүс хоорондоо танилцах холбогдох боломж илүү ихэснэ
- Хэрэглэгч өөрийн шинэ сонирхлоо нээн илрүүлж тэргээрээ дамжуулан өөрийгөө хөгжүүлэх, боломжтой болно
- Хүүхэд залуус чөлөөт цагаа зөв боловсон өнгөрүүлэх сонголтууд нэмэгдэнэ. Мөн хайж олоход хялбар болно
- Өмнө нь олон нийтийн сүлжээ ашиглан эвэнтийн мэдээлэл авдаг байсан хэрэглэгчдийн сэтгэл ханамж нэмэгдэнэ

2. СИСТЕМИЙН ШИНЖИЛГЭЭ

Системийн ерөнхий шаардлага болон шинжилгээ

2.1 Системийн ерөнхий тодорхойлолт

Энэхүү апликашн нь эвэnt зохион байгуулах эвэntэд бүртгүүлэх процессыг нэгдсэн нэг системд төвлөрүүлэн тухайн үйл явцыг илүү хялбарчлах зориулалттай. Эвэntийг хайж олох, бүртгүүлэх, үүсгэх гэх үндсэн үйл ажиллагаатай.

2.1.1 Үйл ажиллагааны тодорхойлолт

- Эвэntийг хайж олох

Хайж олохдоо хэрэглэгчид санал болгож буй хэсгээс сонгох эсвэл хэрэглэгч өөрөө эвэntийн төрөл, газар зүйн байршил зэрэг мэдээллээр шүүлт хийх хайх.

- Эвэntэд бүртгүүлэх

Хэрэглэгч эвэntэд бүртгүүлэхдээ эвэntийг хайж олсны дараа хэрэгтэй мэдээллүүдийг оруулан төлбөрийг төлснөөр эвэntэд бүртгүүлнэ.

- Эвэnt үүсгэх

Хэрэглэгч эхлээд шаардлагатай мэдээллүүдийг оруулан эвэnt үүсгэгчээр бүртгүүлсний дараа эвэnt үүсгэх боломжтой болно. Эвэntийг үүсгэхдээ эвэntийн талаарх дэлгэрэнгүй мэдээлэл болон тасалбарын мэдээллийг оруулан нийтлэнэ.

- Эвэnt зохион байгуулагчийг дагах

Хэрэглэгч ямар нэгэн эвэnt зохион байгуулагчийг дагах боломжтой ба ингэснээрээ тухайн зохион байгуулагчийн дараагийн эвэntүүдийн талаарх мэдээллийг сонордуулга байдлаар болон нүүр хуудасны эхэнд байрлахаар харж болно.

- Хэрэглэгч өөрт таалагдсан эвэntээ тэмдэглэх

Хэрэглэгч өөрт таалагдсан эвэntийг тэмдэглэснээр дараа тухайн эвэнтийн талаарх мэдээллийг өөрийн таалагдсан цэсээс харах боломжтой байна.

2.1.2 Системийн бүрэлдэхүүн хэсгүүд

Нэгж

- Хэрэглэгч / системд бүртгүүлсэн хэрэглэгч
- Үүсгэгч / системд бүртгүүлсэн, үүсгэгчийн мэдээлэл баталгаажсан хэрэглэгч

Процесс

- Эвэnt үүсгэх
- Эвэnt устгах
- Эвэнтэд бүртгүүлэх
- Эвэnt үүсгэгчийг дагах
- Таалагдсан эвэндээ тэмдэглэх
- Тасалбар худалдан авах
- Санхүүгийн мэдээлэл боловсруулж харуулах

2.2 Системийн шаардлага

Шаардлагыг тодорхойлохдоо хэрэглэгч болон зохион байгуулагчийн хувьд функциональ шаардлагуудыг тодорхойлж, функциональ бус шаардлагуудыг ерөнхий шаардлага гэж үзэж тодорхойллоо.

2.2.1 Ерөнхий шаардлага

- ЕШ10 - Системийг олон хэрэглэгч нэгэн зэрэг ашиглах боломжтой байна.
- ЕШ20 - Хэрэглэгч системд хэзээ ч хаанаас ч хандах боломжтой, 24 цаг найдвартай ажилладаг байх
- ЕШ30 - Зөвхөн бүртгэлтэй хэрэглэгчид нэвтрэх ба token шалгаж нэвтрэнэ.
- ЕШ40 - Хэрэглэгчийн интерфэйс дизайн хэрэглэгчид ашиглахад хялбар, ойлгомжтой байх
- ЕШ50 - Алдааны мессеж болон сануулга мессеж бүрийг ойлгомжтой харуулах
- ЕШ60 - Шинэ хэрэглэгч бүртгэгдэхэд JWT ашиглан нууц үгийг encrypt хийж өгөгдлийн санд хадгална.
- ЕШ70 - Ухаалаг утасны бүх үйлдлийн систем дээр ажилладаг байх
- ЕШ80 - Өгөгдөл мэдээллийг аюулгүй найдвартай хадгалагдаг байх

2.2.2 Хэрэглэгчийн функциональ шаардлага

- ХШ10 - Хэрэглэгч шинээр бүртгүүлнэ
- ХШ20 - Хэрэглэгч өөрийн бүртгэлтэй мэйлээр нэвтэрч орно
- ХШ30 - Хэрэглэгч нууц үгээ солих боломжтой байх
- ХШ40 - Хэрэглэгч төлбөр төлөн тасалбар захиалах боломжтой байх
- ХШ50 - Хэрэглэгч төрлөөр хайлт хийх боломжтой байх
- ХШ60 - Хэрэглэгч эвэnt зохион байгуулагчийг дагах боломжтой байх

- ХШ70 - Хэрэглэгч эвэntийн зохион байгуулагдах газрын байршлыг газрын зураг дээр харах боломжтой байх
- ХШ80 - Хэрэглэгч таалагдсан эвэнтээ жагсаалтад хийх боломжтой байх
- ХШ90 - Хэрэглэгч тасалбарын түүхээ харах боломжтой байх
- ХШ100 - Хэрэглэгч эвэнт үүсгэгчийн дэлгэрэнгүй мэдээлэл харах боломжтой байх
- ХШ110 - Хэрэглэгч эвэнт зохион байгуулагч болох боломжтой байх
- ХШ120 - Хэрэглэгч тасалбараа буцаах боломжтой байх.

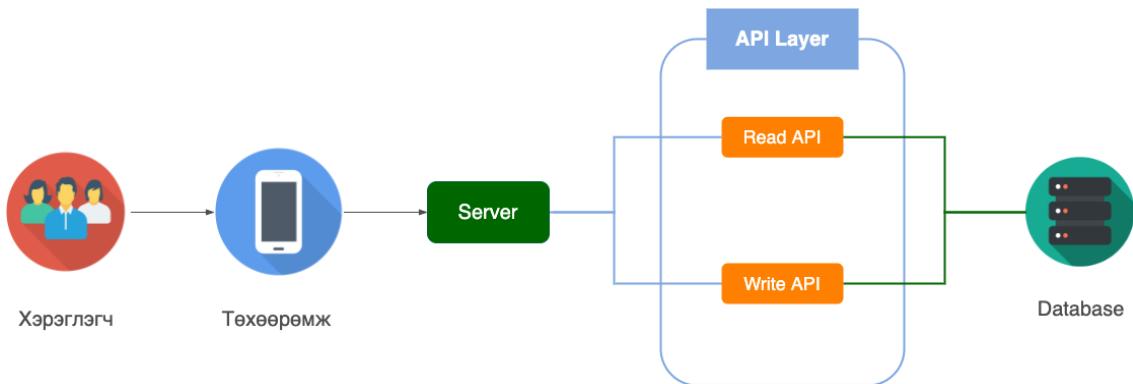
2.2.3 Зохион байгуулагчийн функциональ шаардлага

- ЗБШ10 - Зохион байгуулагч өөрийн бүртгэлтэй мэйлээр нэвтэрч орно
- ЗБШ20 - Зохион байгуулагч эвэнт үүсгэх боломжтой байх
- ЗБШ30 - Зохион байгуулагч өөрийн зохиож байсан эвэнтүүдийн түүхээ харах боломжтой байх
- ЗБШ40 - Зохион байгуулагч тасалбарын борлуулалтын талаарх мэдээллийг харах боломжтой байх
- ЗБШ50 - Зохион байгуулагч эвэнтийн тасалбарыг олон төрлөөр үүсгэх боломжтой байх
- ЗБШ60 - Зохион байгуулагч тасалбарын QR кодыг шалгах боломжтой байх
- ЗБШ70 - Зохион байгуулагч борлуулалтаас бий болсон орлогыг татан авах боломжтой байх
- ЗБШ80 - Зохион байгуулагч үүсгэсэн эвэнтэй цуцлах боломжтой байх.

2.3 Системийн ерөнхий зохиомж

Аппликацийн ерөнхий дизайн зураг нь програм бүхэлдээ хэрхэн яаж ажиллах, хэрэглэгчид хоорондоо ямар хамааралтай ажиллаж буй зэргийг харуулах зорилготой ерөнхий байдлаар загварчилсан диаграмм юм.

Системийн ерөнхий диаграмм



Зураг 2.1: Ерөнхий диаграмм

2.4 Өгөгдлийн урсгалын шинжилгээ

2.4.1 Системийн оролт

Хэрэглэгч болох өгөгдөл

- Овог, нэр
- Майл хаяг

Эвэnt зохион байгуулагч болох өгөгдөл

- Зохион байгуулагчийн нэр
- Мэдээлэл
- Зохион байгуулах төрөл

- Банкны дансны дугаар

Эвэнт үүсгэх өгөгдөл

- Эвэнтийн нэр

- Тайлбар

- Төрөл

- Төлөв

- Тасалбар

- Үнэ

- Тоо ширхэг

- Төрөл

- Байршил

Эвэндэт бүртгүүлэх өгөгдөл

- Утасны дугаар

- Тасалбарын ширхгийн тоо

2.4.2 Системийн гаралт

- Үүсгэсэн эвэнтууд

- Тасалбарууд

- Тасалбарын QR код

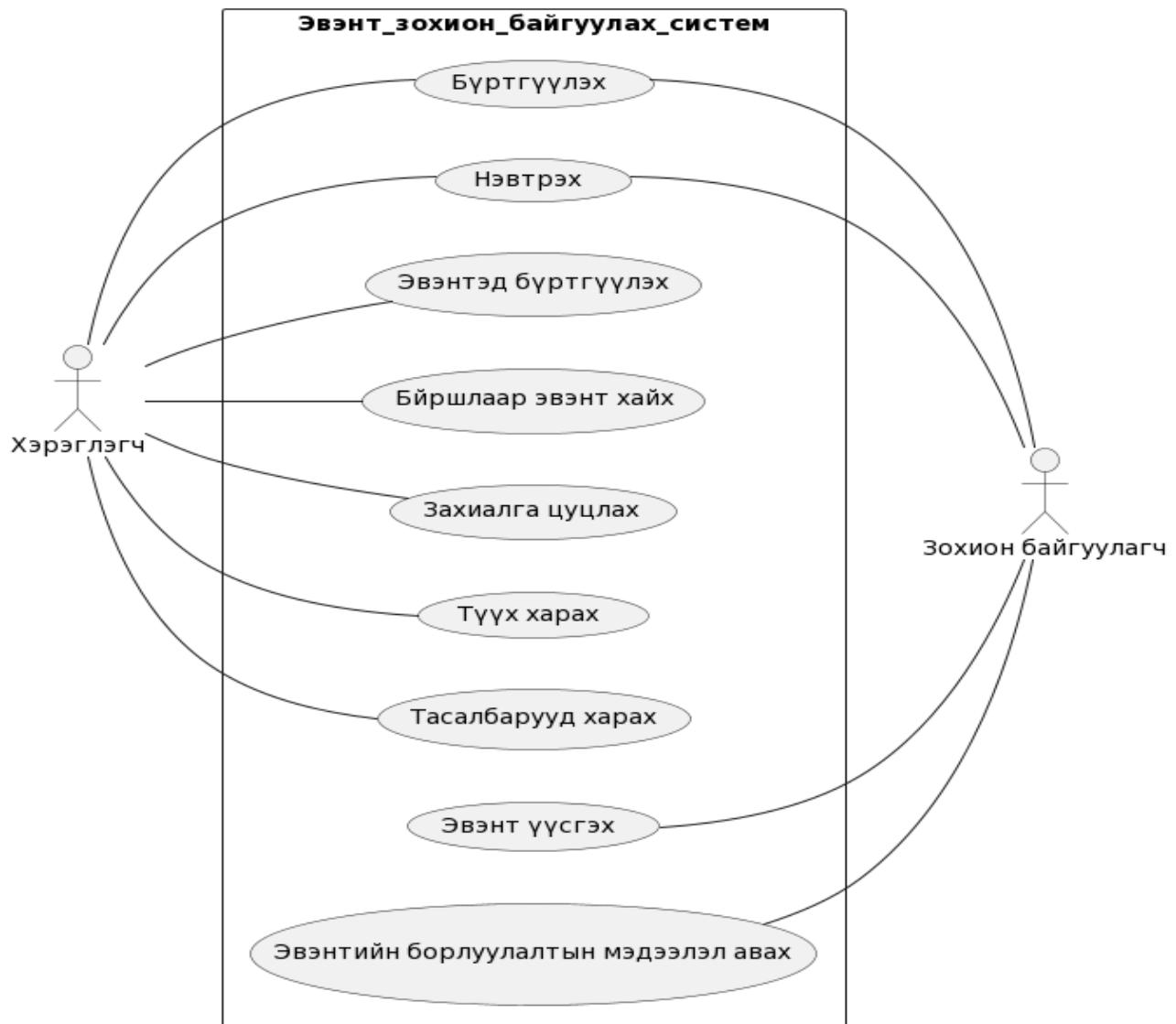
- Санхүү болон тасалбар борлуулалтын дүн мэдээ анализ

- Эвэнтийн байршлын мэдээлэл

2.5 Объект хандлагат шинжилгээ

2.5.1 Use case диаграмм

Use-case диаграмм нь тухайн систем дээр ямар ямар үйлдлүүд хийгдэх, ямар хэрэглэгч юу юуг хийх вэ гэдгийг харуулах буюу товчхондоо систем юу хийж чадах вэ гэдгийг бүтэцлэн харуулсан диаграмм юм.

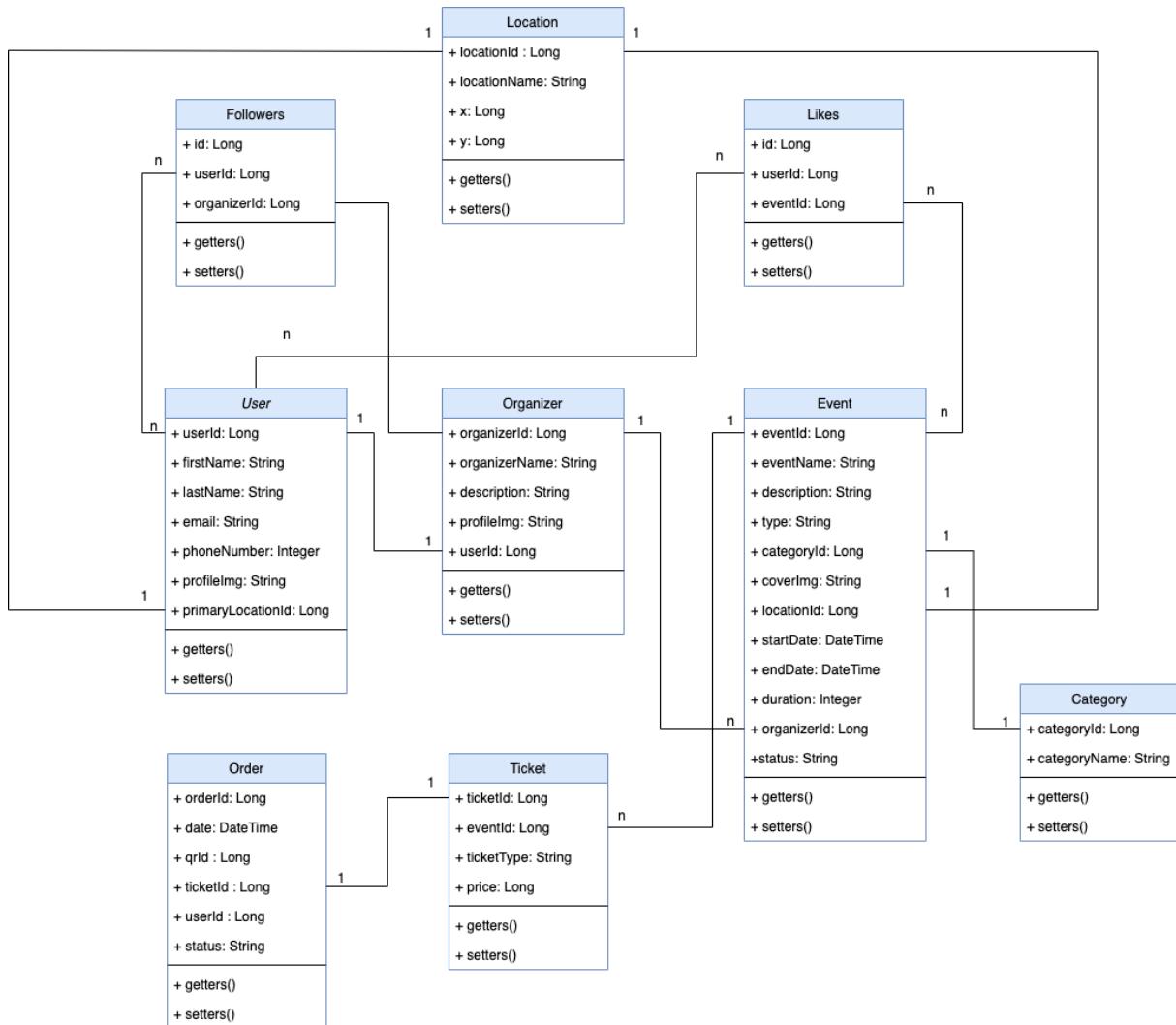


Зураг 2.2: Use case диаграмм

2.5.2 Бүтцийн диаграм

Бүтцийн диаграм буюу класс диаграм нь системийг эсвэл түүний аль нэг модулийг объект хандлагат програмчлалын түвшинд дүрслэн харуулдаг диаграм буюу программын статик төлөвийг дурсэлдэг.

Класс диаграм нь системийн объектуудын шинж чанар буюу атрибут болон хийх үйлдэл функциуудыг агуулсан байна. Класс диаграмыг дүрсэлснээр программын кодыг бичихэд илүү ойлгомжтой байдаг.



Зураг 2.3: Класс диаграм

Классын объектууд

- Хэрэглэгч (User)
- Зохион байгуулагч (Organizer)
- Дагагчид (Followers)
- Таалагдсан эвэнтүүд (Likes)
- Эвэnt (Event)
- Эвэнтийн төрөл (Category)
- Тасалбар (Ticket)
- Захиалга (Order)
- Байршил (Location) эдгээр объектуудыг классын гишүүн өгөгдөл болон getter setter функцын хамтаар диаграмд дүрсэлсэн болно.

2.6 Өгөгдлийн сангийн зохиомж

2.6.1 Өгөгдлийн толь бичиг

Хэрэглэгч , зохион байгуулагч, байршил, эвэнтийн төрөл, тасалбар, дагагчид, таалагдсан эвэнтүүд, захиалга, эвэnt гэх эдгээр хүснэгтүүд өгөгдлийн сан дээр үүснэ. Өгөгдлийн сангийн зохиомжид дүрсэлсэн объект тус бүрээр өгөгдлийн тайлбарыг зургаар оруулав.

Хэрэглэгч

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
userId (primary key)	Дугаар (давтагдахгүй)	Long
firstname	Нэр	String
lastname	Овог	String
email	майл хаяг	String
phonenumber	утасны дугаар	Integer
profileImg	нүүр зураг	String
primaryLocationId (foreign key)	байршлын дугаар	Long

Зураг 2.4: Хэрэглэгч - өгөгдлийн тайлбар

Зохион байгуулагч

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
organizerId (primary key)	Дугаар (давтагдахгүй)	Long
organizerName	нэр	String
description	тайлбар	String
profileImg	нүүр зураг	String
userId(foreign key)	хэрэглэгчийн дугаар	Long

Зураг 2.5: Зохион байгуулагч - өгөгдлийн тайлбар

Захиалга

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
orderId (primary key)	Дугаар (давтагдахгүй)	Long
date	Үүссэн огноо	Date
QRid	QR кодын дугаар	Long
ticketId(foreign key)	тасалбарын дугаар	Long
userId(foreign key)	хэрэглэгчийн дугаар	Long
status	төлөв	String

Зураг 2.6: Захиалга - өгөгдлийн тайлбар

Хэрэглэгч эвэntийн тасалбарыг худалдаж авах үед энэ хүснэгтэд шинэ мөр нэмэгдэх ба энэ мөр нь дээрх мэдээллийг агуулна.

Төрөл

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
Id (primary key)	Дугаар (давтагдахгүй)	Long
categoryName	төрлийн нэр	String

Зураг 2.7: Эвэntийн төрөл - өгөгдлийн тайлбар

Эвэнт

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
eventId (primary key)	Дугаар (давтагдахгүй)	Long
eventName	Эвэнтийн нэр	String
description	Эвэнтийн тайлбар	String
type	төрөл	String
categoryId (foreign key)	Эвэнтийн категорийн дугаар	Long
coverImg	эвэнтийн зураг	String
locationId	байршлын дугаар	Long
startDate	эхлэх огноо	Date
endDate	дуусах огноо	Date
duration	эвэнтийн үргэлжлэх хугацаа	Integer
organizerId (foreign key)	зохион байгуулагчийн дугаар	Long
status	төлөв	String

Зураг 2.8: Эвэнт - өгөгдлийн тайлбар

Дагагчид

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
Id (primary key)	Дугаар (давтагдахгүй)	Long
userId(foreign key)	хэрэглэгчийн дугаар	Long
organizerId (foreign key)	зохион байгуулагчийн дугаар	Long

Зураг 2.9: Дагагчид - өгөгдлийн тайлбар

Таалагдсан эвэнтүүд

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
Id (primary key)	Дугаар (давтагдахгүй)	Long
userId(foreign key)	хэрэглэгчийн дугаар	Long
eventId(foreign key)	эвэнтийн дугаар	Long

Зураг 2.10: Таалагдсан эвэнтүүд - өгөгдлийн тайлбар

Байршил

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
locationId (primary key)	Дугаар (давтагдахгүй)	Long
locationName	Байршилын нэр	String
x	уртрага	Long
y	өргөрөг	Long

Зураг 2.11: Байршил - өгөгдлийн тайлбар

Тасалбар

Талбарын нэр	Тайлбар	Өгөгдлийн төрөл
ticketId (primary key)	Дугаар (давтагдахгүй)	Long
eventId (foreign key)	эвэнтийн дугаар	Long
ticketType	төрөл	String
price	үнэ	Integer

Зураг 2.12: Тасалбар - өгөгдлийн тайлбар

3. СИСТЕМИЙН ЗОХИОМЖ

3.1 Интерфэйсийн зохиомж

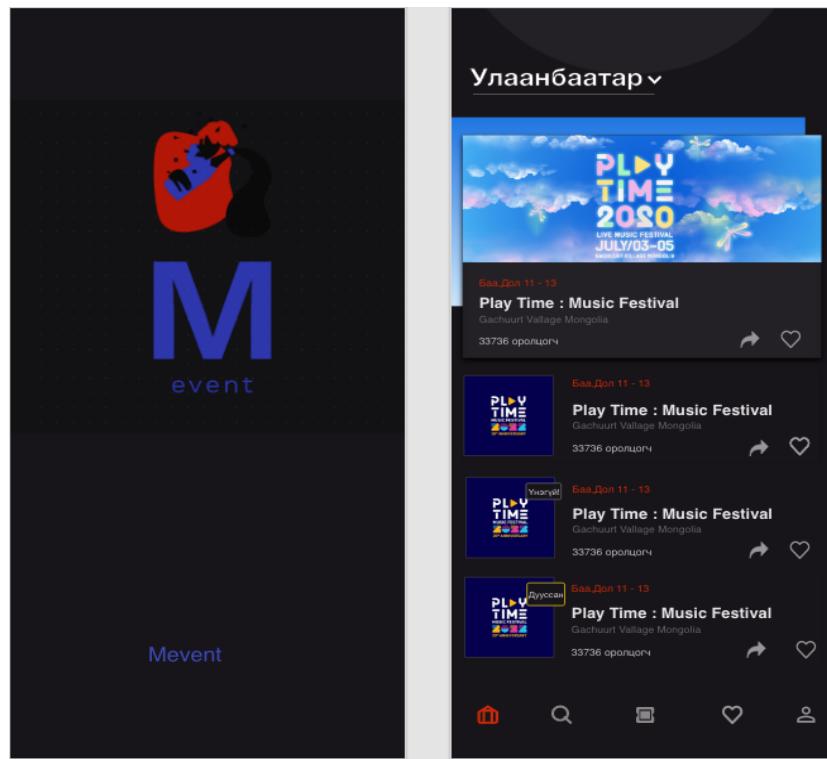
Хөгжүүлэлт хийхээс өмнө дэлгэцийн ерөнхий зохиомжийг гаргахын тулд үндсэн зарим компонентүүдийн харагдах байдлыг хийсвэрээр дүрсэлсэн зургийг **AdobeXD** ашиглан зурсан. Ингэснээр хэрэглэгчийн интерфэйсийг хөгжүүлэхэд илүү хялбар, ойлгомжтой байлгах зорилготой юм.

Аппликашны дэлгэцийн бүтэц

- Аппликашн ажиллаж эхлэхэд харагдах хэсэг
- Бүртгүүлэх/Нэвтрэх хэсэг
- Нуур хуудас
- Нэг эвэntийн талаарх дэлгэрэнгүй мэдээлэл харуулах хуудас
- Зохион байгуулагчийн мэдээллийг харуулах хуудас
- Хэрэглэгчийн таалагдсан эвэнтүүд болон дагасан зохион байгуулагчдыг харуулах хуудас
- Хэрэглэгчийн худалдан авсан тасалбарын мэдээллийг харуулах хуудас
- Хайлт хийх хэсэг
- Хэрэглэгчийн хувийн мэдээллийг хэсэг
- Эвэnt үүсгэх хэсэг
- Эвэнтийн борлуулалтын мэдээлэл харах хэсэг
- QR код унших хэсэг

Аппликашн ажиллаж эхлэхэд харагдах хэсэг болон нүүр хуудас

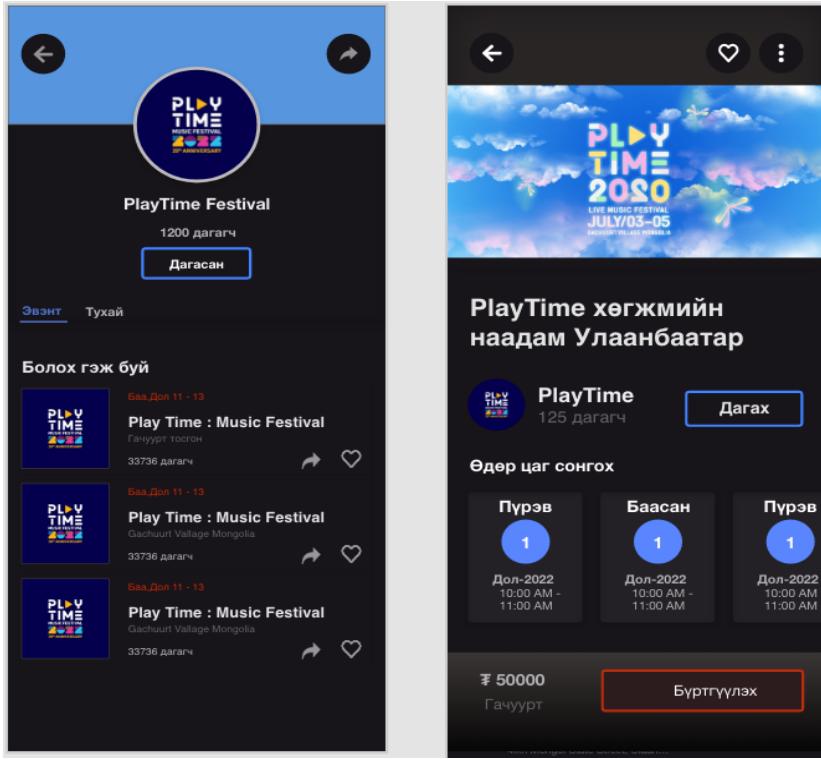
- Хэрэглэгчийн дагасан зохион байгуулагчдын зохиож буй эвэнтүүдийг харуулах.
- Тусгай эвэнтийн мэдээллийг том байдлаар харуулах



Зураг 3.1: Аппликашн ажиллаж эхлэхэд харагдах хэсэг болон нүүр хуудас

Эвэнтийн дэлгэрэнгүй мэдээлэл болон зохион байгуулагчийн хэсэг

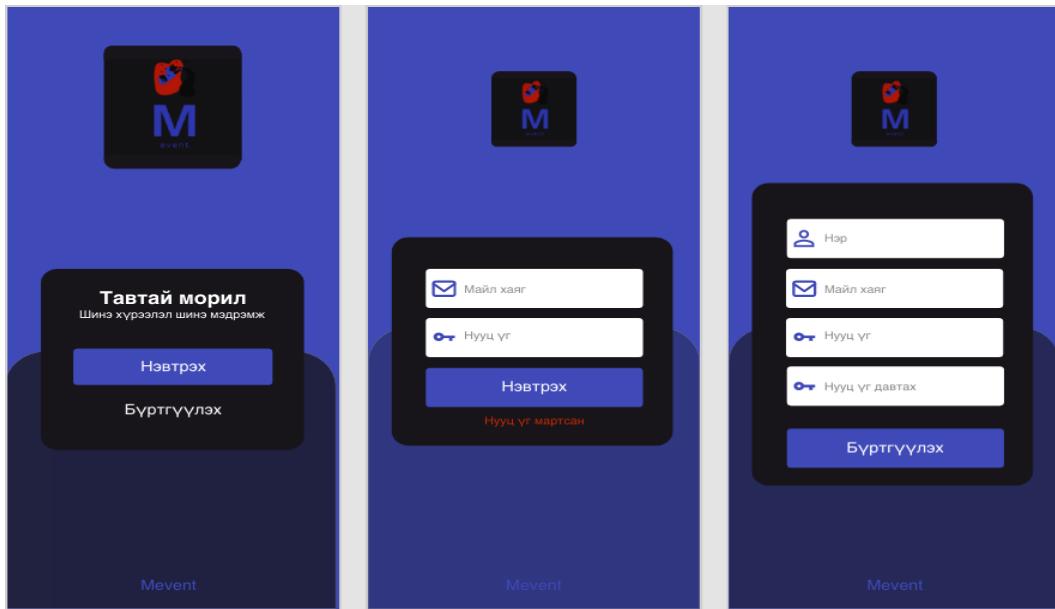
- Зохион байгуулагчийн эвэнтүүдийн түүхийг харуулах
- Апп-ыг хэрэглэж буй хэрэглэгчийн тухайн зохион байгуулагчийн дагаж буй эсэхийг харуулах.
- Эвэнтийн үнэ, дэлгэрэнгүй болон бусад мэдээллүүдийг харуулах



Зураг 3.2: Эвэnt болон зохион байгуулагч

Бүртгүүлэх болон нэвтрэх хэсэг

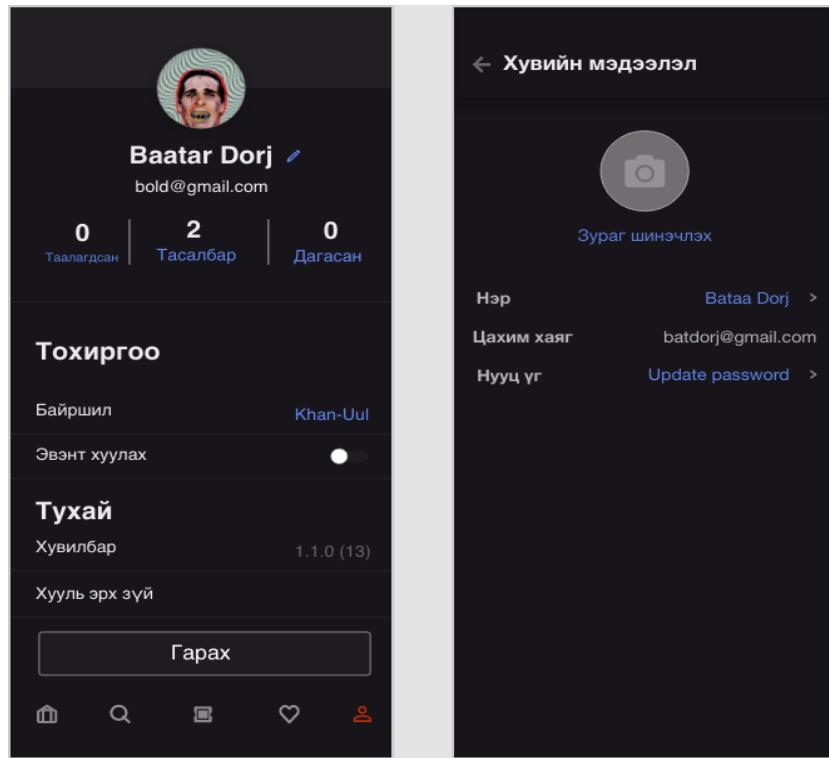
- Майл хаяг нь дахин давтагдахгүй байна.
- Нууц үг нь том жижиг үсэг агуулсан зургаагаас дээш тэмдэгт байна.



Зураг 3.3: Нэвтрэх болон бүртгүүлэх хэсэг

Хэрэглэгчийн хувийн мэдээлэл

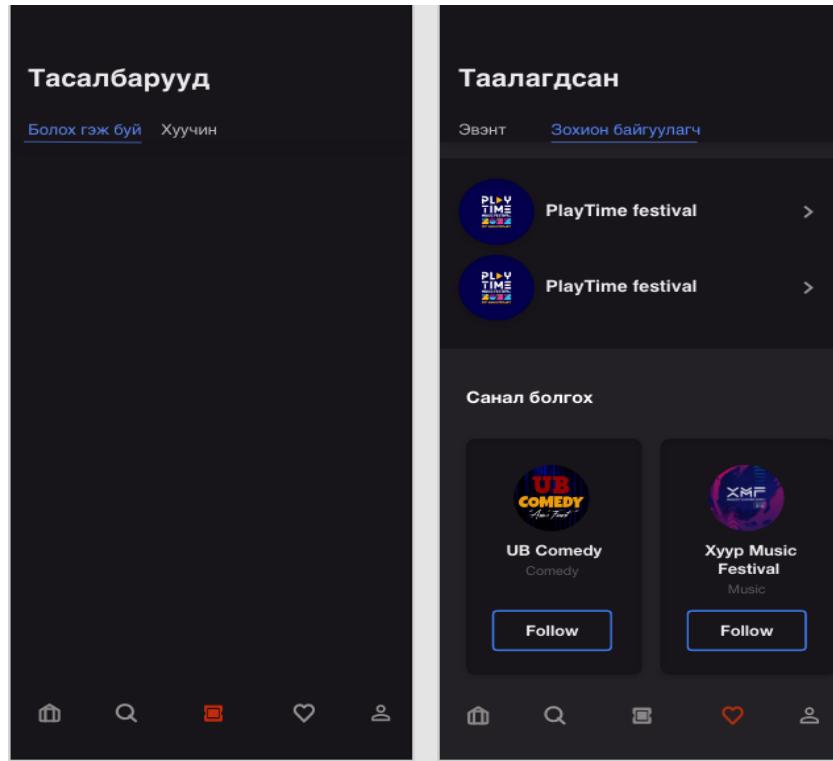
- Хэдэн зохион байгуулагч дагасан болон хэдэн эвэнт таалагдсан мэдээллүүдийг тоог харах
- Аппликацийны хувилбарыг харах.



Зураг 3.4: Хэрэглэгчийн хэсэг

Худалдаж авсан тасалбар болон таалагдсан эвэntүүдийг харуулах хэсэг

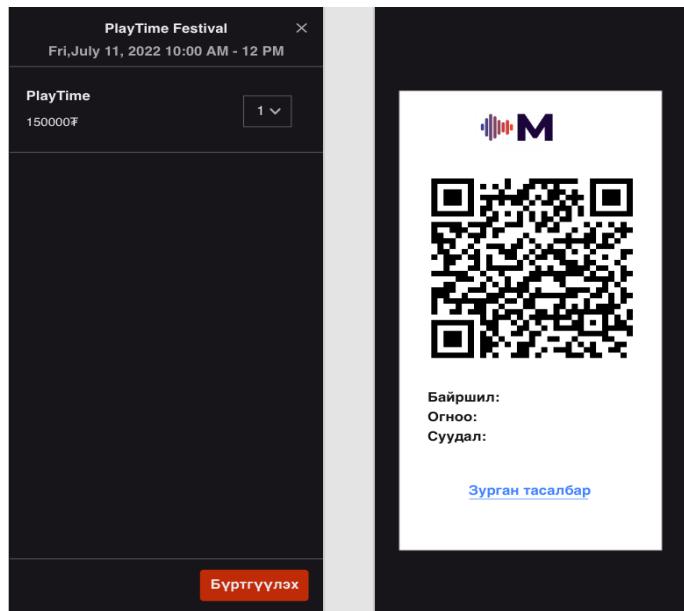
- Худалдаж авсан тасалбаруудын түүх хугацааны дарааллаар харагдана.
- Өөрийн таалагдсан эвэнтийн дэлгэрэнгүй мэдээллийг сум дээр дарснаар харна



Зураг 3.5: Тасалбар болон таалагдсан

Эвэнтэд бүртгүүлэх болон худалдаж авсан тасалбарын QR кодын хэсэг

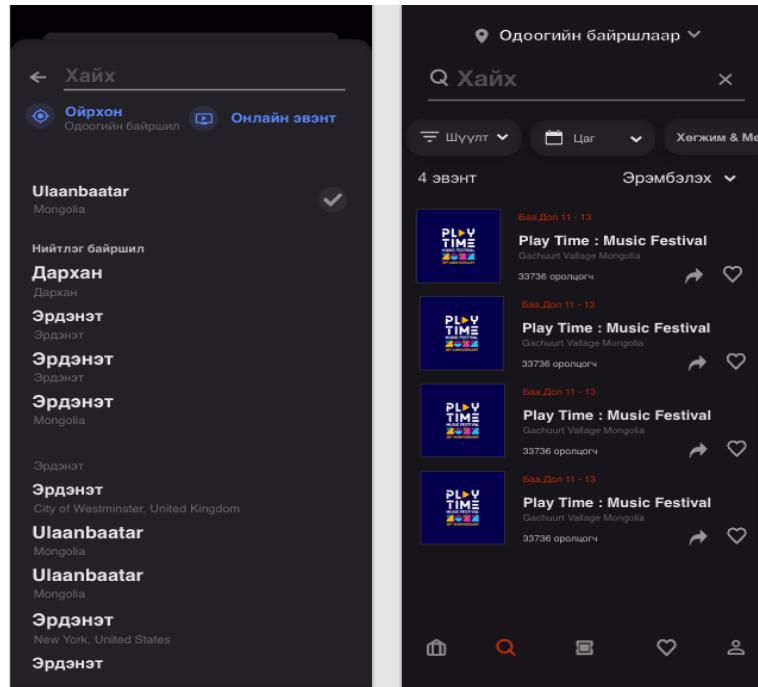
- Бүртгүүлэх товч дээр дарснаар эвэнтэд оролцох эрхтэй болно.
- QR кодны доод хэсэгт тасалбарын төрөл болон хугацааны мэдээлэл, дээд хэсэгт эвэнтийн нэр багтана.



Зураг 3.6: Тасалбар болон qr код

Хайлтын хэсэг

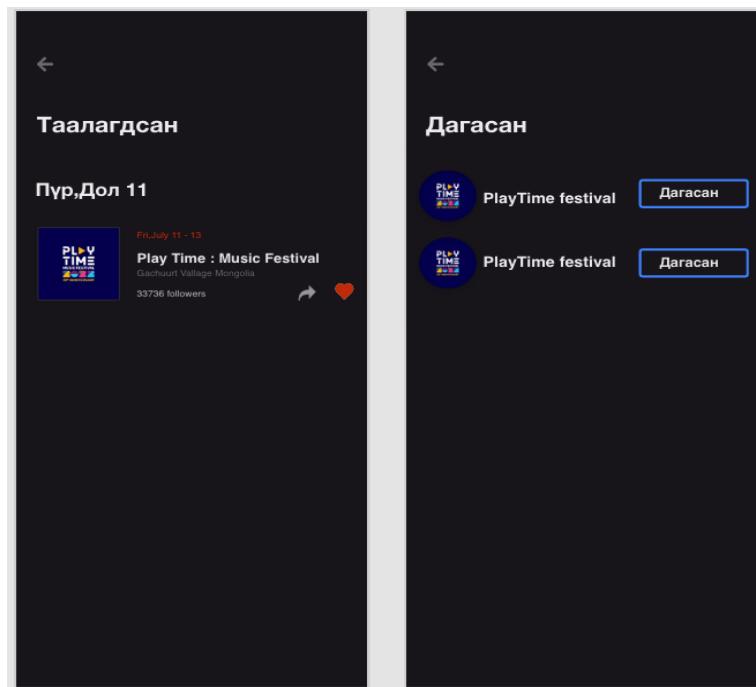
- Эвэнтийн төрөл хэсэг дээр дарснаар тухайн төрлөөр хайлт хийн үр дүнг харуулна.
- Сонгосон төрлөөс түлхүүр үгээр хайлт хийж болно.



Зураг 3.7: Хайлтын хэсэг

Хэрэглэгчийн дагасан зохион байгуулагчдыг харуулах хэсэг

- Дагасан зохион байгуулагчийн ерөнхий мэдээллийг харах ба зураг дээр дарснаар дэлгэрэнгүй мэдээллийг харна.



Зураг 3.8: Дагасан зохион байгуулагчдыг харуулах хэсэг

4. ХЭРЭГЖҮҮЛЭЛТ

Энэ хэсэгт сонгон авсан сэдвийн дагуу бэлтгэл ажлуудыг хийж гүйцэтгэсний дараа үндсэн хөгжүүлэлтийн ажлыг хэрхэн хийснийг харуулна. Хэсэг бүрд ямар шаардлагын дагуу хийж гүйцэтгэснийг тэмдэглэв.

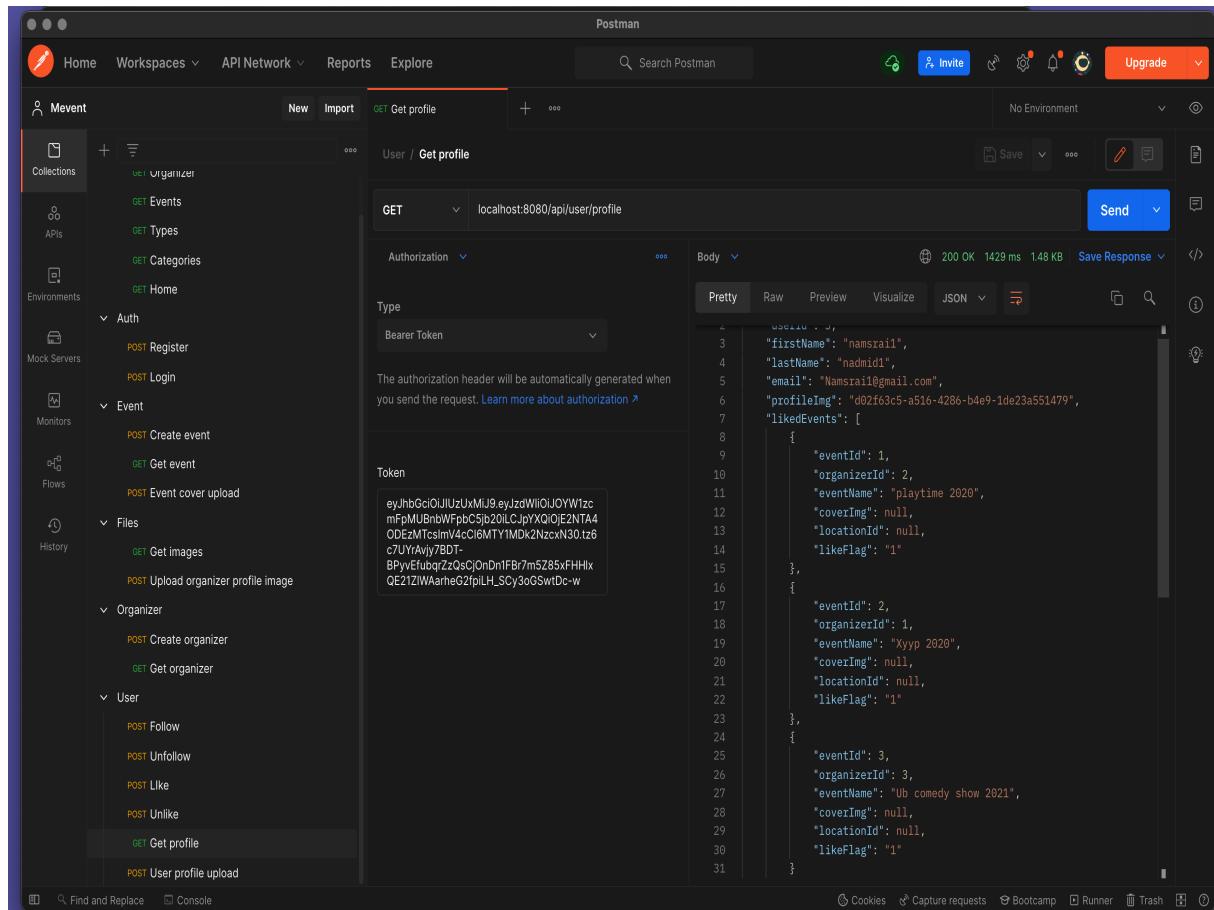
4.1 Системийн хөгжүүлэлт

Хэрэглэгчийн интерфэйсийг харуулах клиент талыг React Native санг ашиглан хөгжүүлж байна. Хэрэглэгчийн интерфэйс дээр харуулах мэдээллийг REST API дуудаж харуулах ба мөн хэрэглэгчийн оруулсан өгөгдөл датаг бааз руу хадгалахдаа post, put гэх мэт HTTP хүсэлтээр дамжуулан шаардлагатай API-г дуудаж хадгалж байна. Харин логик үйлдлүүдийг гүйцэтгэх сервер талын хөгжүүлэлтийг хийхдээ Spring Boot фреймворк дээр REST API сервисүүдийг хөгжүүлсэн.

4.1.1 Хөгжүүлсэн API жагсаалт

- Хэрэглэгч бүртгэх API - Sign up new user API
- Хэрэглэгч нэвтрэх API - Login user API
- Зөвхөн нэг хэрэглэгчийн мэдээллийг авах - Get User By Id
- Хэрэглэгч зохион байгуулагч болох - Create Organizer
- Эвэnt үүсгэх - Create Event
- Эвэнтийн тасалбар худалдаж авах - Buy Ticket
- Хэрэглэгч зохион байгуулагчийг дагах - Follow
- Хэрэглэгч эвэнтийг таалагдсан гэж бүртгэх - Like
- Хэрэглэгч өөрийн зургийг оруулах хэсэг - Upload Profile

- Эвэntийн зураг оруулах - Upload Event Cover
- Эвэнтийн мэдээлэл авах - Get Event By Id
- Аппликацийны нүүр хуудсанд харуулах мэдээллийг авах - Get Home



Зураг 4.1: Хөгжүүлсэн API жагсаалт

4.2 Бүртгүүлэх болон нэвтрэх хэсэг

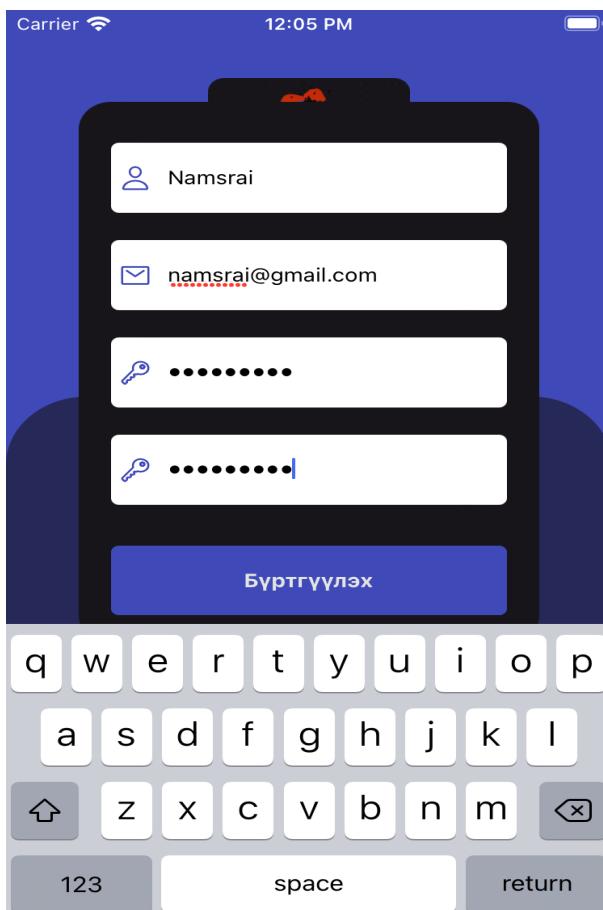
Биелгдсэн шаардлагууд

- XIII10 - Хэрэглэгч шинээр бүртгүүлнэ
- XIII20 - Хэрэглэгч өөрийн бүртгэлтэй мэйлээр нэвтэрч орно

- ЕШ30 - Зөвхөн бүртгэлтэй хэрэглэгчид нэвтрэх ба token шалгаж нэвтрэнэ
- ЕШ60 - Шинэ хэрэглэгч бүртгэгдэхэд JWT ашиглан нууц үгийг encrupt хийж өгөгдлийн санд хадгална.

4.2.1 Бүртгүүлэх

Хэрэглэгч бүртгүүлэхдээ шаардлагатай мэдээллийг харгалзах талбаруудад оруулж бүртгүүлэх товчийг дарснаар бүртгэл баталгаажиж өгөгдлийн санд мэдээлэл хадгалагдана.



Зураг 4.2: Бүртгүүлэх

Өгөгдлийн санд хадгалагдах байдал Хэрэглэгчийн үүсгэсэн бүртгэлийн мэдээлэл өгөгдлийн санд дараах байдлаар хадгалагдана. Хадгалагдахаа нууц үг нь шифрлэгдсэн тусгай

ТЭМДЭГТ болно.

Data Output							Explain	Messages	Notifications
	user_id [PK] bigint	email character varying (20)	first_name character varying (20)	last_name character varying (20)	password character varying (120)	profile_img character varying (255)		location_id bigint	
1	1	bataa@gmail.com	bataa	myagaa	\$2a\$10\$90kQJfMdnlQciFF1QLSG0msPGc/VZLjORmZmeVHatME2JrPWPatK	[null]		[null]	
2	2	Namsrai@gmail.com	namsrai	nadmid	\$2a\$10\$pa14B4zL0hJHIBVNWH.r.2bPLBs3mrsq4z/oisE4qC3r8povsu	[null]		[null]	
3	3	Namsrai1@gmail.com	namsrai1	nadmid1	\$2a\$10\$njpf5gV1Pe3F3KCPh3iWNu2CLf/q3n0gcCgkcUO88CrdrqdtlZFO	d02f63c5-a516-4286-b4e9-1de23a551479		[null]	

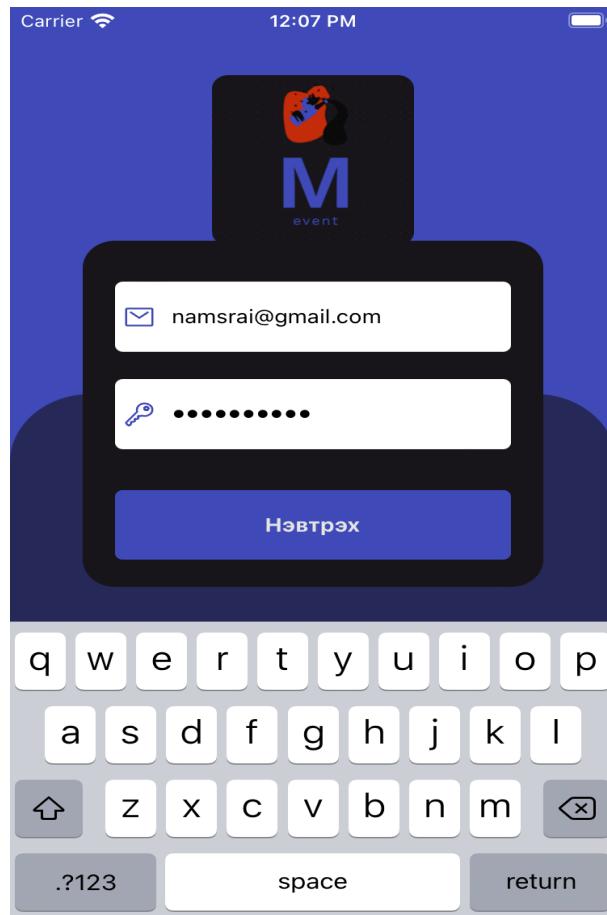
Зураг 4.3: Бүртгүүлэх - Өгөгдлийн санд

Амжилтгүй болох нөхцөлүүд

- Төхөөрөмж интернэтэд холбогдоогүй байх
- Оруулсан майл хаяг давхардах
- Давтан оруулсан нууц үг алдаатай байх

4.2.2 Нэвтрэх

Хэрэглэгч нэвтрэхдээ шаардлагатай мэдээллийг харгалзах талбаруудад оруулж нэвтрэх товчийг дарснаар аппликеишнд нэвтрэн API сервисээс буцаасан token - ийг төхөөрөмж дээрээ хадгалан дараагийн үйлдүүд хийхдээ ашиглана.



Зураг 4.4: Нэвтрэх

```
LOG {"email": "Namsrai@gmail.com", "firstName": "namsrai", "lastName": "nadmid1", "token": "eyJhbGciOiJIUzI1njc1b20iLCJpYXQiOjE2NTA5NDY5NDQsImV4cCI6MTY1MTAzMzM0Nj0.94ViBInvzENjRFoHUC6LdpXZjeaXEkp9fx3t0errFUqt3jkb1fidhujrT73lphTK09ZG907ceWQN9nS-w", "type": "Bearer", "userId": 3}
```

Зураг 4.5: Нэвтрэх - API буцаалт

Амжилтгүй болох нөхцөлүүд

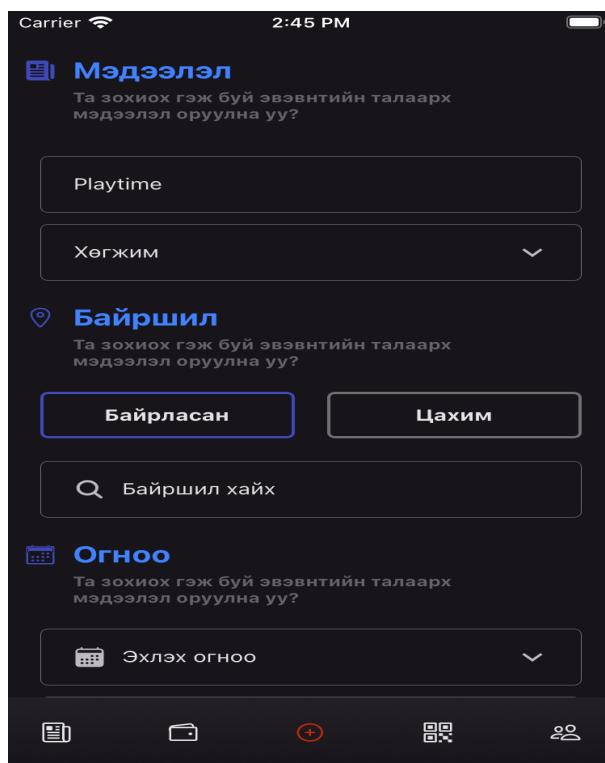
- Төхөөрөмж интернэтэд холбогдоогүй байх
- Оруулсан майл хаяг болон нууц үг буруу байх

4.3 Эвэнт үүсгэх

Биелэгдсэн шаардлагууд

- ЗБШ20 - Зохион байгуулагч эвэnt үүсгэх боломжтой байх
- ЗБШ30 - Зохион байгуулагч өөрийн зохиож байсан эвэнтүүдийн түүхээ харах боломжтой байх
- ЗБШ50 - Зохион байгуулагч эвэнтийн тасалбарыг олон төрлөөр үүсгэх боломжтой байх

Зохион байгуулагч эвэнтийг үүсгэхдээ эвэнтийн талаарх мэдээллүүдийг харгадзах талбарт оруулан хадгалах товч дарснаар хүсэлт илгээгдэнэ.



Зураг 4.6: Эвэнт үүсгэх

Амжилтгүй болох нөхцөлүүд

- Төхөөрөмж интернэтэд холбогдоогүй байх
- Дутуу мэдээлэл оруулах

- Эвэнтийн нэр давхцах

Өгөгдлийн санд үүсэх өгөгдөл

	event_id [PK] bigint	cover_img character varying (255)	event_name character varying (50)	category_id bigint	type_id bigint	location_id bigint	organizer_id bigint
1	1 [null]		playtime 2020	1	2	[null]	2
2	2 [null]		Xhyp 2020	2	1	[null]	1
3	3 [null]		Ub comedy show 2021	1	2	[null]	3

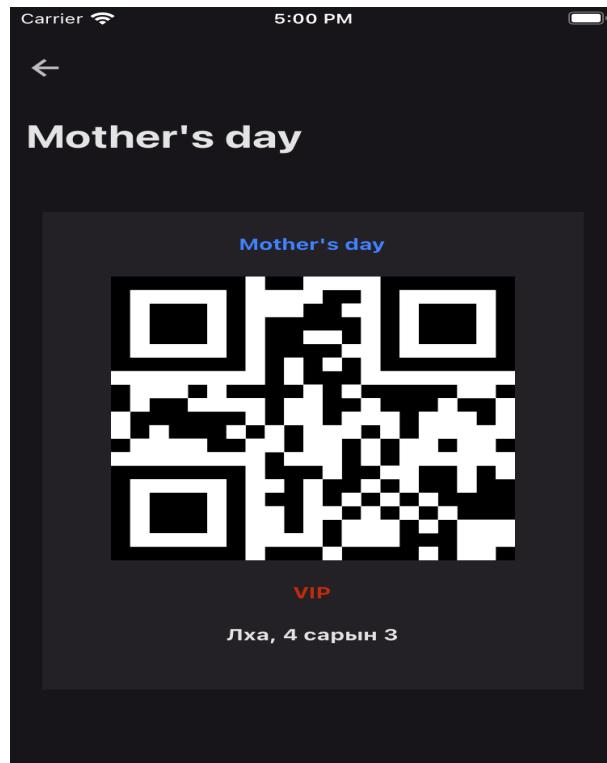
Зураг 4.7: Эвэнт үүсгэх - Өгөгдлийн санд

4.3.1 Tasalbar xudaldan avah

Биелгдсэн шаардлагууд

- ХШ40 - Хэрэглэгч төлбөр төлөн тасалбар захиалах боломжтой байх
- Зохион байгуулагч тасалбарын борлуулалтын талаарх мэдээллийг харах боломжтой байх
- ЗБШ60 - Зохион байгуулагч тасалбарын QR кодыг шалгах боломжтой байх

Хэрэглэгч эвэнтийн дэлгэрэнгүй мэдээлэл харах хэсгийн бүртгүүлэх хэсэг дээр дарсны дараа худалдан авах хэсэг рүү шилжин орж худалдаж авах тасалбарын тоо болон өдрийг сонгоод баталгаажуулах товчийг дарснаар худалдан авалт хийгдэн хэрэглэгчид тасалбарын зургийг буцаан харуулна.



Зураг 4.8: Тасалбар худалдан авах

Үйлдэл амжилттай болсны дараа өгөгдлийн сангийн Order хүснэгтэд шинэ мөр нэмэгдэнэ.

	order_id [PK] bigint	date timestamp without time zone	status character varying (255)	ticket_id bigint	user_id bigint
1	1	2022-04-25 04:05:06	active	2	1

Зураг 4.9: Тасалбар худалдан авах - Өгөгдлийн санд

Амжилтгүй болох нөхцөлүүд

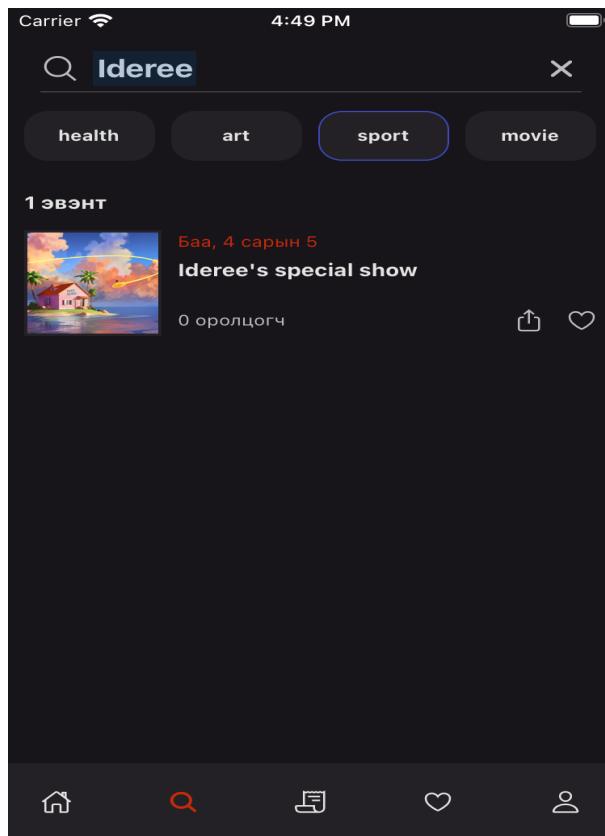
- Тасалбарын ширхэгийн тоо оруулаагүй байх
- Өдөр сонгоогүй байх
- Тасалбар дууссан байх

4.3.2 Хайлт хийх

Биелэгдсэн шаардлагууд

- ХШ50 - Хэрэглэгч төрлөөр хайлт хийх боломжтой байх

Хэрэглэгч төрлөөр болон түлхүүр үгээр хайлтыг хийн үр дүнг харна.



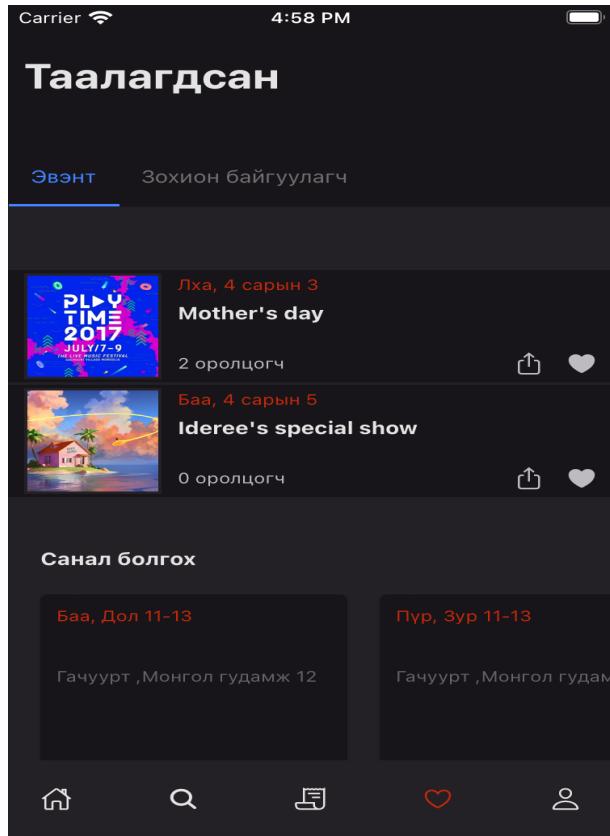
Зураг 4.10: Хайлт хийх

4.3.3 Дагасан болон таалагдсан

Биелэгдсэн шаардлагууд

- ХШ80 - Хэрэглэгч таалагдсан эвэнтээ жагсаалтад хийх боломжтой байх
- ХШ100 - Хэрэглэгч эвэнт үүсгэгчийн дэлгэрэнгүй мэдээлэл харах боломжтой байх

- ХШ60 - Хэрэглэгч эвэнт зохион байгуулагчийг дагах боломжтой байх



Зураг 4.11: Таалагдсан болон дагасан

ДҮГНЭЛТ

Миний бие энэхүү бакалаврын судалгааны ажлын хүрээнд эвэnt зохион байгуулах аппликацийн хөгжүүлэх зорилго тавин ажиллалаа. Энэхүү зорилгодоо хүрэхийн тулд программ хангамж зохиох үе шатуудыг судлан түүнийхээ дагуу зорилтууд тавин шаардлагын шинжилгээ боловсруулж, системийн ерөнхий зохиомжийг гарган төгсгөлд нь хөгжүүлэлтийн ажлыг **React Native , Java Spring Boot , Postgresql** - ашиглан хийж гүйцэтгэлээ.

Цаашид бакалаврын судалгааны ажлын хүрээнд хийж гүйцэтгэсэн аппликационээ улам сайжруулан хөгжүүлж зах зээлд нийлүүлэх төлөвлөгөөтэй байна.

Bibliography

- [1] <https://www.meetup.com/>
- [2] <https://reactnative.dev/docs>
- [3] <https://reactnavigation.org/>
- [4] <https://github.com/react-native-video/react-native-video>
- [5] <https://reactnavigation.org/>

A. КОДЫН ХЭРЭГЖҮҮЛЭЛТ

Кодын хэсэгт клиент болон сервер талын хэрэгжүүлэлтээс зарим жишээ кодуудыг орууллаа.

A.1 Back-side хэрэгжүүлэлтийн код

A.1.1 Хэрэглэгч бүртгүүлэх, нэвтрэх APIs

AuthController.java

```
1  @CrossOrigin(origins = "*", maxAge = 3600)
2  @RestController
3  @RequestMapping("/api/auth")
4  public class AuthController {
5
6      @Autowired
7      AuthenticationManager authenticationManager;
8
9      @Autowired
10     UserRepository userRepository;
11
12     @Autowired
13     PasswordEncoder encoder;
14
15     @Autowired
16     JwtUtils jwtUtils;
17
18     @PostMapping("/signin")
19     public ResponseEntity<?> authenticateUser(@Valid @RequestBody
20         LoginRequest loginRequest) {
21
22         Authentication authentication =
23             authenticationManager.authenticate(
24                 new UsernamePasswordAuthenticationToken(
25                     loginRequest.getEmail(),
26                     loginRequest.getPassword()
27                 )
28             );
29
30         SecurityContextHolder.getContext().setAuthentication(authentication
31             );
32
33         String jwt = jwtUtils.generateJwtToken(authentication);
34         UserDetailsImpl userDetails = (UserDetailsImpl) authentication.
35             getPrincipal();
36
37         return ResponseEntity.ok(
38             new JwtResponse(
39                 jwt,
40                 userDetails.getUserId(),
41                 userDetails.getFirstName(),
```

```

40         userDetails.getLastName(),
41         userDetails.getEmail()
42     )
43 );
44 }
45
46 @PostMapping("/signup")
47 public ResponseEntity<?> registerUser(@Valid @RequestBody
48     SignupRequest signUpRequest) {
49
50     if (userRepository.existsByEmail(signUpRequest.getEmail())) {
51         return ResponseEntity
52             .badRequest()
53             .body(
54                 new MessageResponse(
55                     "Error: Email is already in use!"
56                 )
57             );
58     }
59
60     // Create new user's account
61     User user = new User(
62         signUpRequest.getFirstName(),
63         signUpRequest.getLastName(),
64         signUpRequest.getEmail(),
65         encoder.encode(signUpRequest.getPassword()
66     );
67
68     userRepository.save(user);
69     return ResponseEntity
70         .ok(new MessageResponse("User registered successfully!"));
71     }
72 }
```

Код A.1: Хэрэглэгч бүртгүүлэх нэвтрэх

A.1.2 Хэрэглэгч бүртгүүлэх болон нэвтрэх үед нууцлалыг хэрэгжүүлэх

JwtUtils.java

```

1  @Component
2  public class JwtUtils {
3      private static final Logger logger = LoggerFactory.getLogger(JwtUtils
4          .class);
5
6      @Value("${mevent.app.jwtSecret}")
7      private String jwtSecret;
8
9      @Value("${mevent.app.jwtExpirationMs}")
10     private int jwtExpirationMs;
```

```

10
11     public String generateJwtToken(Authentication authentication) {
12
13         UserDetailsImpl userPrincipal = (UserDetailsImpl) authentication.
14             getPrincipal();
15
16         return Jwts.builder()
17             .setSubject(
18                 userPrincipal.getEmail())
19             .setIssuedAt(new Date())
20             .setExpiration(
21                 new Date(
22                     new Date()
23                         .getTime() + jwtExpirationMs))
24             .signWith(SignatureAlgorithm.HS512, jwtSecret)
25             .compact();
26
27     public String getUserNameFromJwtToken(String token) {
28         return Jwts.parser()
29             .setSigningKey(jwtSecret)
30             .parseClaimsJws(token)
31             .getBody()
32             .getSubject();
33     }
34
35     public Long getUserIdFromToken(){
36         Authentication authentication = SecurityContextHolder
37             .getContext()
38             .getAuthentication();
39
40         UserDetailsImpl userDetails = (UserDetailsImpl) authentication.
41             getPrincipal();
42         return userDetails.getUserId();
43     }
44
45     public boolean isUserAuthenticated(){
46
47         Authentication authentication = SecurityContextHolder
48             .getContext()
49             .getAuthentication();
50         return null != authentication && !"anonymousUser").equals(
51             authentication.getName());
52     }
53
54     public boolean validateJwtToken(String authToken) {
55         try {
56             Jwts.parser().setSigningKey(jwtSecret).parseClaimsJws(authToken);
57             return true;
58         } catch (SignatureException e) {
59             logger.error("Invalid JWT signature: {}", e.getMessage());
60         } catch (MalformedJwtException e) {

```

```

59     logger.error("Invalid JWT token: {}", e.getMessage());
60 } catch (ExpiredJwtException e) {
61     logger.error("JWT token is expired: {}", e.getMessage());
62 } catch (UnsupportedJwtException e) {
63     logger.error("JWT token is unsupported: {}", e.getMessage());
64 } catch (IllegalArgumentException e) {
65     logger.error("JWT claims string is empty: {}", e.getMessage());
66 }
67     return false;
68 }
69 }
```

Код А.2: Хэрэглэгч бүртгүүлэх болон нэвтрэх үед нууцлалыг хэрэгжүүлэх

A.1.3 Зураг оруулахтай холбоотой API

FileController.java

```

1  @RestController
2  @CrossOrigin(origins = "*", maxAge = 3600)
3  @RequestMapping("/api/image")
4  @AllArgsConstructor
5  public class FileController {
6      FileStorageServiceImp storageService;
7      FileDBRepository fileDBRepository;
8      JwtUtils jwtUtils;
9
10     @PostMapping("/upload")
11     public ResponseEntity<MessageResponse> uploadFile(@RequestParam("file") MultipartFile file) {
12         String message = "";
13         try {
14             storageService.store(file);
15             message = "Uploaded the file successfully: " + file.
16                 getOriginalFilename();
17             return ResponseEntity.status(HttpStatus.OK).body(new
18                 MessageResponse(message));
19         } catch (Exception e) {
20             message = "Could not upload the file: " + file.
21                 getOriginalFilename() + "!";
22             return ResponseEntity.status(HttpStatus.EXPECTATION_FAILED)
23                 .body(new MessageResponse(message));
24         }
25     }
26
27     //upload organizer profile
28     @PostMapping("/user/upload")
29     public ResponseEntity<?> uploadUserProfileImg(@RequestParam("file")
30             MultipartFile file){
31         try{
```

```

27     MessageResponse response = storageService.uploadUserProfile
28         (file);
29     return ResponseEntity
30         .ok()
31         .body(response);
32 } catch (Exception e) {
33     e.printStackTrace();
34     return ResponseEntity
35         .status(HttpStatus.EXPECTATION_FAILED)
36         .body(new MessageResponse("Upload error"));
37 }
38
39 //upload event cover
40 @PostMapping("/event/{eventId}/upload")
41 public ResponseEntity<?> uploadEventCoverImg(@RequestParam("file")
42     MultipartFile file, @PathVariable("eventId") Long eventId){
43     try{
44         MessageResponse response = storageService.uploadEventCover(
45             eventId, file);
46         return ResponseEntity
47             .ok()
48             .body(response);
49     } catch (Exception e) {
50         e.printStackTrace();
51         return ResponseEntity
52             .status(HttpStatus.EXPECTATION_FAILED)
53             .body(new MessageResponse("Upload error"));
54     }
55
56 //upload organizer profile
57 @PostMapping("/organizer/{organizerId}/upload")
58 public ResponseEntity<?> uploadOrganizerProfileImg(@RequestParam("
59     file") MultipartFile file , @PathVariable("organizerId") Long
60     organizerId){
61     try{
62         MessageResponse response = storageService.
63             uploadOrganizerProfile(organizerId, file);
64         return ResponseEntity
65             .ok()
66             .body(response);
67     } catch (Exception e) {
68         e.printStackTrace();
69         return ResponseEntity
70             .status(HttpStatus.EXPECTATION_FAILED)
71             .body(new MessageResponse("Upload error"));
72     }
73
74 //Download image

```

```

73     @GetMapping("/download/{id}")
74     public ResponseEntity<byte[]> getFile(@PathVariable String id) {
75         FileDB fileDB = storageService.getFile(id);
76         return ResponseEntity.ok()
77             .header(HttpHeaders.CONTENT_DISPOSITION, "attachment;
78                 filename=\"" + fileDB.getName() + "\"")
79             .body(fileDB.getData());
80     }
81
82     //get image
83     @GetMapping("/{id}")
84     public ResponseEntity<byte[]> fromDatabaseAsResEntity(@PathVariable
85         ("id") String id)
86         throws SQLException {
87
88         Optional<FileDB> userImage = fileDBRepository.findById(id);
89         byte[] imageBytes = null;
90         if (userImage.isPresent()) {
91
92             imageBytes = userImage
93                 .get()
94                 .getData();
95         }
96
97     }

```

Код А.3: Зураг оруулахтай холбоотой API

A.1.4 Эвэнттэй холбоотой API

EventController.java

```

1     @PostMapping("/create")
2     public ResponseEntity<?> createEvent(@Valid @RequestBody EventDto
3         eventDto){
4         if(eventRepository.existsByEventName(eventDto.getEventName())){
5             return ResponseEntity
6                 .badRequest()
7                 .body(
8                     new MessageResponse(
9                         "Error: Event name is already in
10                            use !"
11                     )
12                 );
13
14     EventType eventType = eventTypeRepository
15         .findById(eventDto.getTypeId())

```

```

15     .orElseThrow();
16
17     Category category = categoryRepository
18         .findById(eventDto.getCategoryId())
19         .orElseThrow();
20
21     Organizer organizer = organizerRepository
22         .findById(eventDto.getOrganizerId())
23         .orElseThrow();
24
25     Event event =
26         new Event(
27             eventDto.getEventName(),
28             organizer,
29             category,
30             eventType
31         );
32
33     eventRepository.save(event);
34     return ResponseEntity
35         .ok()
36             .new MessageResponse("Event created successfully
37             !"));
38
39     @GetMapping("/{eventId}")
40     ResponseEntity<?> getEvent(@Valid @PathVariable("eventId") Long
41     eventId){
42         if (!eventRepository.existsById(eventId)){
43             return ResponseEntity
44                 .ok()
45                 .body(new MessageResponse("empty"));
46         }
47         else{
48             return ResponseEntity
49                 .ok()
50                 .body(
51                     eventServiceImp.getEvent(eventId)
52                 );
53         }
54     }

```

Код А.4: Эвэнттэй холбоотой API

A.1.5 Эвэнтийн api- аар дамжуулах авах өгөгдлийг бэлдэж өгөх

EventServiceImp.java

```

1     @Service
2     @AllArgsConstructor
3     public class EventServiceImp implements EventService{

```

```

4
5     LikesRepository likesRepository;
6     EventRepository eventRepository;
7     FollowersRepository followersRepository;
8     JwtUtils jwtUtils;
9
10    @Override
11    public MessageResponse createEvent(EventDto eventDto) {
12        return null;
13    }
14
15    @Override
16    public boolean isEventLiked(Long eventId, Long userId) {
17        return likesRepository
18            .existsByUserUserIdAndEventEventId(
19                userId,
20                eventId
21            );
22    }
23
24    @Override
25    public EventResponse getEvent(Long eventId) {
26        Event event = eventRepository
27            .findById(eventId)
28            .orElseThrow();
29
30        return new EventResponse(
31            event.getEventId(),
32            event.getOrganizer().getOrganizerId(),
33            event.getEventName(),
34            event.getCoverImg(),
35            null,
36            isEventLiked(
37                eventId,
38                jwtUtils.getUserIdFromToken()) ? "1" : "0"
39        );
40    }
41    @Override
42    public List<EventResponse> getEvents() {
43        List<Event> events = eventRepository.findAll();
44        List<EventResponse> eventResponses = new ArrayList<>();
45
46        Long loggedUserId = 0L;
47
48        if (jwtUtils.isUserAuthenticated()) {
49            loggedUserId = jwtUtils.getUserIdFromToken();
50        }
51        for (Event e : events){
52            boolean isUserLiked = false;
53
54            if (loggedUserId > 0){
55                isUserLiked = likesRepository

```

```

56         .existsByUserUserIdAndEventEventId(
57             loggedUserId,
58             e.getEventId()
59         );
60     }
61
62     EventResponse r = new EventResponse(
63         e.getEventId(),
64         e.getOrganizer().getOrganizerId(),
65         e.getEventName(),
66         e.getCoverImg(),
67         null,
68         isUserLiked ? "1" : "0"
69     );
70     eventResponses.add(r);
71 }
72
73 return eventResponses;
74 }
75 @Override
76 public List<EventResponse> getEventsByFollowers() {
77     if (!jwtUtils.isUserAuthenticated()) {
78         return null;
79     }else{
80         Long loggedUserId = jwtUtils.getUserIdFromToken();
81         List<Followers> followedData = followersRepository
82             .findAllByUserUserId(loggedUserId);
83         List<Event> eventsByFollowedOrganizers = new ArrayList<>();
84
85         for(Followers followers: followedData){
86             eventsByFollowedOrganizers
87                 .addAll(
88                     eventRepository
89                         .findAllByOrganizerOrganizerId(
90                             followers
91                                 .getOrganizer()
92                                 .getOrganizerId()
93                                     ()
94
95         );
96     List<EventResponse> responses = new ArrayList<>();
97
98     for ( Event e : eventsByFollowedOrganizers){
99         EventResponse eventResponse = new EventResponse(
100             e.getEventId(),
101             e.getOrganizer().getOrganizerId(),
102             e.getEventName(),
103             e.getCoverImg(),
104             null,
105             isEventLiked(
106                 loggedUserId

```

```

107             ,e.getEventId()
108         ) ? "1" : "0"
109     );
110     responses.add(eventResponse);
111 }
112 return responses;
113 }
114 }
115
116 @Override
117 public List<EventResponse> getLikedEvents() {
118     List<EventResponse> events = this.getEvents();
119     List<EventResponse> likedEvents = new ArrayList<>();
120
121     for (EventResponse e : events){
122         if (e.getLikeFlag().equals("1")){
123             likedEvents.add(e);
124         }
125     }
126     return likedEvents;
127 }
128 }
```

Код А.5: Эвэntийн api- аар дамжуулах авах өгөгдлийг бэлдэж өгөх

A.2 Client-side хэрэгжүүлэлтийн код

A.2.1 Аппликацийны нүүр хэсгийг харуулах

HomeScreen.js

```

1 import {
2     View,
3     FlatList,
4     SafeAreaView,
5     StyleSheet,
6     Image,
7     useWindowDimensions,
8     ScrollView,
9     TouchableOpacity,
10 } from 'react-native';
11 import React, {useEffect} from 'react';
12 import {useNavigation, useTheme} from '@react-navigation/native';
13 import {IconButton} from '../common/icon-button';
14 import {bottomSheet} from '../components/bottom-sheet';
15 import {layout, size, text} from '../styles';
16 import {CText} from '../common/c-text';
17 import {events} from '../demoData';
18 import {RedText} from '../common/red-text';
19 import {BlackText} from '../common/black-text';
20 import {GreyText} from '../common/grey-text';
21 import {EventItem} from '../components/event-item';
```

```

22 import {EventCard} from '../components/event-card';
23
24 export const HomeScreen = () => {
25   const {width} = useWindowDimensions();
26   const {navigate} = useNavigation();
27   const {colors} = useTheme();
28
29   useEffect(() => {
30     return () => {};
31   }, []);
32
33   const Header = () => (
34     <View
35       style={[styles.header, {width: width / 2.5, borderColor: colors.
36         border}]}>
37       <CText style={[text.lgThin, {marginRight: size.md}]}></CText
38       <IconButton icon="chevron-down" onPress={() => bottomSheet.show()}
39         />
40     </View>
41   );
42
43   const Banner = ({e}) => (
44     <View style={styles.banner}>
45       <Image
46         source={e.img}
47         blurRadius={20}
48         style={[styles.blurredBanner, {width: width - size.lg}]}>
49       />
50       <TouchableOpacity
51         style={[styles.bannerBox, {backgroundColor: colors.card}]}
52         onPress={() => navigate('Event', {event: e})}
53         activeOpacity={0.9}>
54         <Image
55           source={e.img}
56           style={[styles.bannerImg, {width: width - size.lg}]}>
57       />
58       <View style={styles.detail}>
59         <RedText>{e.date}</RedText>
60         <View>
61           <CText numberOfLines={1} style={text.md}>
62             {e.title}
63           </CText>
64           <BlackText numberOfLines={1}>{e.location}</BlackText>
65         </View>
66         <View style={layout.hsb}>
67           <GreyText>123 followers</GreyText>
68           <View style={styles.buttonBar}>
69             <IconButton icon="md-share-outline" s />
70             <IconButton icon="heart-outline" s />
71           </View>
72         </View>
73       </View>
74     );
75
76   return (
77     <EventCard
78       title="Event"
79       date="2023-09-22"
80       location="New York, NY"
81       img="https://example.com/event-image.jpg"
82       followers="123"
83       share="Share"
84       like="Like" />
85   );
86 }

```

```

71         </View>
72     </TouchableOpacity>
73   </View>
74 );
75
76 const Suggest = () => (
77   <View style={[styles.suggest, {backgroundColor: colors.card}]}>
78     <CText style={[text.md, {marginVertical: size.lg}]}>      </CText
79       >
80     <FlatList
81       horizontal
82       data={events}
83       renderItem={({item}) => <EventCard event={item} />}
84       keyExtractor={item => item.id}
85     />
86   </View>
87 );
88
89 return (
90   <SafeAreaView style={styles.root}>
91     <View style={[styles.circle, {backgroundColor: colors.card}]} />
92     <ScrollView contentContainerStyle={styles.scrollView}>
93       <Header />
94       <Banner e={events[1]} />
95       <View style={{marginTop: size.lg + 20}}>
96         {events.map(el => (
97           <EventItem
98             data={el}
99             key={el.id}
100            onPress={() => navigate('Event', {event: el})}
101          />
102        )));
103      </View>
104      <Suggest />
105      <Banner e={events[3]} />
106      <View style={{marginTop: size.lg + 20}}>
107        {events.map(el => (
108          <EventItem
109            data={el}
110            key={el.id}
111            onPress={() => navigate('Event', {event: el})}
112          />
113        )));
114      </View>
115    </ScrollView>
116  </SafeAreaView>
117 );
118
119 const styles = StyleSheet.create({
120   root: {
121     flex: 1,

```

```
122 },
123   circle: {
124     width: 500,
125     height: 500,
126     position: 'absolute',
127     left: -250,
128     top: -250,
129     borderRadius: 250,
130   },
131   header: {
132     ...layout.hsb,
133     borderBottomWidth: StyleSheet.hairlineWidth,
134     paddingVertical: size.sm,
135     marginTop: size.md,
136     marginHorizontal: size.md,
137   },
138   banner: {
139     height: 200,
140     marginVertical: size.lg,
141   },
142   blurredBanner: {
143     height: 200,
144   },
145   bannerBox: {
146     height: 200,
147     left: size.md,
148     top: size.lg + 10,
149     position: 'absolute',
150     ...layout.shadow,
151   },
152   bannerImg: {
153     height: 100,
154   },
155   detail: {
156     marginLeft: size.md,
157     marginTop: size.md,
158     justifyContent: 'space-between',
159     flex: 1,
160   },
161   buttonBar: {
162     ...layout.hsb,
163     width: 60,
164     margin: size.sm,
165   },
166   scrollView: {
167     paddingBottom: 100,
168   },
169   suggest: {
170     paddingHorizontal: size.md,
171     marginTop: size.lg,
172     paddingBottom: size.lg,
173   },
```

174 });

Код А.6: Аппликашны нүүр хэсгийг харуулах