公立はこだて未来大学 2021 年度 システム情報科学実習
グループ報告書

Future University Hakodate 2021 Systems Information Science Practice

Group Report

プロジェクト名

ビーコン IoT で函館のまちをハックする - Beacon FUN Rejuvenation

Project Name

Leverage the Beacon IoT for Our Smarter Life in Hakodate Real Downtown - Beacon FUN Rejuvenation

グループ名

All

Group Name

All

プロジェクト番号/Project No.

7

プロジェクトリーダー/Project Leader

なし　None

グループリーダー/Group Leader

なし　None

グループメンバー/Group Member

佐々木紀明　Noriaki Sasaki　柿本翔大　Shodai Kakimoto
大巻佑太　Yuta Omaki　中山寿似也　Juniya Nakayama
小田嶋亜美　Ami Odashima　山口将馬　Shoma Yamaguchi
石田海祐　Kaiyu Ishida　山内彩土　Sayato Yamauchi
田村修吾　Shugo Tamura　萩原啓道　Hiromichi Hagiwara
山本竜生　Ryuki Yamamoto

指導教員

松原克弥　藤野雄一　鈴木昭二　奥野拓　鈴木恵二

Advisor

Katsuya Matsubara　Yuichi Fujino　Sho'ji Suzuki　Taku Okuno　Keiji Suzuki

提出日

2022 年 1 月 19 日

Date of Submission

January 19, 2022

概要

本プロジェクトは，函館の街のさまざまな場所にビーコンを設置し，新しい価値を創造するサービスの考案と開発を目的とする．新たに函館の街なかにビーコンを設置するだけでなく，すでに未来大学内に設置されているビーコンを用いたサービスの考案も視野に入れた．サービスを提案するにあたりフィールドワークを実施し，函館の街や観光が抱える課題を発見した．その結果から，複数のサービスアイデアを創出した．さらに，「ビーコンである理由」「函館らしいサービス」「サービスの新規性」「サービスのおもしろさ」を評価基準として定め，サービスアイデアを 3 つに絞り込んだ．その後，サービスアイデアごとにチームを分け，サービスアイデアを具体化，開発体制を決定し，開発に取り掛かった．各チームで相互にレビューをしながら開発を進め，自分だけの水族館を作り朝市を楽しく巡られる「あさいち水族館」，音楽共有で重く話しにくい雰囲気を変える「Favor」，学内の友人の居場所を知ることができる「FLAT」の 3 サービスを開発した．

キーワード　ビーコン，函館，街や観光が抱える課題

（文責: 柿本翔大）

Abstract

We aim to design and develop unique services that create new value with beacons colocated around the city of Hakodate. We also considered to use beacons put in our university not just installing new beacons in city of Hakodate. In order to propose services, we conducted fieldwork to discover the issues facing the city and tourism in Hakodate. Based on the results, we came up with several service ideas. In addition, we decided to narrow down the service ideas to three based on the following evaluation criteria: “why we use beacon”, “whether the service seems Hakodate”, “novelty of the service”, and “interest of the service”. After that, we were divided into three teams for each service idea, concretized the service ideas, decided the development system, and started development. We progressed our development with each team reviewed each other’s work, and developed three services: “Asaichi Aquarium”, which allows users to create their own aquarium and enjoy visiting morning markets; “Favor”, which changes the heavy and difficult to talk atmosphere by sharing music; and “FLAT”, which allows users to find out where their friends are in FUN.

Keyword Beacon, Hakodate, Issues facing the city and tourism in Hakodate

（文責: 柿本翔大）

目次

第Ⅰ部

活動の概要

第 1 章　本プロジェクトの背景と目的

1.1　背景

現在，IoT(Internet of Things) を導入する企業が増加している．IoT とは，従来インターネットに接続されていなかったさまざまなモノに通信機能を持たせ，モノとインターネットを通信できるようにするしくみのことである．多くの企業の目的は，データ収集・解析をし，効率化を図ることである．実際に，導入した企業の 7 割が，IoT 導入効果を実感している [1]．情報を発信する IoT デバイスの 1 つに，ビーコンがある．ビーコンは，一定の範囲に電波を発信する．主にプッシュ通知，位置情報，行動履歴を利用したサービスに用いられる．ビーコンに対応するデバイスとして，スマートフォンがある．スマートフォンの人口に対する保有率は 2019 年時点で，8 割を超えている [2]．対応するデバイス数の増加に伴い，サービスの需要の増加は高まっている．

函館市は，特徴が多く存在する．歴史的資源であったり，周りが海であることや，自然の多さから地域資源の豊富さ等といった点が挙げられる．以上に挙げた特徴のように，函館は魅力の多い街である．

（文責: 萩原啓道）

1.2　ビーコンについて

ビーコンとは，電波を送信し，その電波をほかの機器で受信ことにより情報を取得するためのものである．本プロジェクトでは特に「Bluetooth Low Energy」（BLE）を利用した BLE ビーコンをビーコンと呼ぶ．BLE とは，Bluetooth の規格で，低消費電力が特徴である [3]．BLE ビーコンの導入事例として，Coca Cola 社の「Coke ON」というサービスがある [4]．このサービスは，Coke ON に対応している自動販売機の近くで，スマートフォンにインストールした専用アプリケーションからクーポンを受け取れるというものだ．自動販売機に BLE ビーコンが搭載されており，スマートフォンが BLE ビーコンの電波の受信範囲に入ると，アプリケーションで検知できるというしくみになっている．

BLE を利用した技術の 1 つに Apple 社が開発した iBeacon というものがある [5]．iBeacon に対応したビーコンが発する情報に，UUID，Major，Minor，RSSI がある．iOS の場合，iBeacon の仕様上ビーコンを認識するためには UUID が必要で，Major と Minor の組み合わせによって，各ビーコンを識別する．RSSI は信号強度を表す数値で，ビーコンからのおおよその距離を測るために用いる．

（文責: 山本竜生）

1.3 目的

本プロジェクトの目的は，函館の街なかに設置したビーコンを用いて，新しい価値を創造するサービスの考案と開発することである．

（文責: 萩原啓道）

第 2 章　本プロジェクトの特徴

2.1　活動体制

2.1.1　プロジェクトリーダーを置かない

本プロジェクトでは，プロジェクトリーダーのみに責任やタスクを集中させないことを目的とし，前年度を踏襲してプロジェクトリーダーを設置しなかった．プロジェクトリーダーを置かないかわりにメンバーそれぞれが責任を持ってプロジェクトに取り組めるよう，ファシリテーターの交代制度や大臣制度を取り入れた．しかし，制度の中でいくつか問題が発生し，特定のメンバーにタスクや責任が集中してしまうこともあった．2.1.2 節で述べる問題はわかりやすい例である．この問題はチームの自主性や責任意識の低さによるものだと考えられる．当初の目的は達成できたと考えるが，この制度の採用には十分検討する必要があったと言える．

（文責: 山内彩土）

2.1.2　ファシリテーター

本プロジェクトの活動では，前期の途中までメンバーが週替わりで 2 名ずつファシリテーターを務めた．理由としては，誰がファシリテーターに適任か判断できなかったためである．メンバー全員が一度ずつ経験した後に，誰がファシリテーターを担当するか話し合った．話し合いの結果，大巻が立候補したことにより，その後の活動のファシリテーターが決定した．この際，ファシリテーターの補佐として大巻が柿本・山内を指名したことにより補佐も決定した．補佐の役割は，ファシリテーターが欠席した際に代わりを務めることや，進行方法を決定する際の助言である．ファシリテーターが決定した後は，活動日の前日に「ファシリ会議」と称して，Discord を用いて補佐とともに次の活動内容についての話し合いをした．そこで，活動内容・時間配分・進行方法を決定し，実際に活動を進めた．しかし，ファシリ会議の際にファシリテーターを 1 人に決定したことにより発生した大きな問題が発覚した．ファシリテーターとは関係のない仕事もしなくてはならなくなってしまったことである．代表的なものとしては，「活動内容の決定」が挙げられる．本来，ファシリテーターは参加者に発言を促したり，話の流れをまとめる人のことを言うため，活動内容の決定は仕事に含まれていない．もちろん，活動内容の決定の際に，意見を出したりすることはあるが，事前にファシリテーターが決めて活動するのは本プロジェクトにおけるファシリテーターの責任範囲外である．問題が発生した理由として，「週替わりで行っていた際は活動内容のほとんどが前回からの引き継ぎだったため考える必要がなかった」ことが挙げられる．つまり，メンバー全員に「活動内容の決定」に対する責任意識がなかったことである．問題を解決するために，「ファシリ会議にほかのメンバーも参加をしても良いと伝える」ことや「次の活動までの宿題として YWT 法を用いたふりかえりをする」といった方法を取ったが，人が集まらなかったり，ふりかえり忘れが多く，あまり効果的とは言えない結果となった．

（文責: 大巻佑太）

2.1.3 議事録

本プロジェクトの活動では，チーム，グループ単位での活動の際に議事録を取った．議事録を取り入れた目的は，プロジェクト内での決定事項の共有，そして，備忘録である．議事録は，チームの中で週替わりに，1 名ずつ担当していった．

（文責: 萩原啓道）

2.1.4 大臣制度

本チームには 2.1.1 節で述べたように，プロジェクトリーダーが存在していない．さまざまな作業の説明責任を分担することを目的として，大臣制度を設けた．たとえば，報告書の作業を取りまとめる人を「報告書大臣」と呼び，報告書大臣は，下書きの提出日や，レビュー日などの日程を調整し，進捗などをレビュアーである担当教員や TA に説明する責任を負った．

各メンバーが自主的に行動し，効率的に活動することを目的として導入したが，本チームにおいては，自主性が足りず，多くの場合，効果的ではなかった．

たとえば，成果発表会の動画制作を担当した動画大臣や，同じく成果発表会のポスター制作を担当したポスター大臣は，うまく作業を分散させることができず，各大臣が実行責任を負う形になってしまった．すなわち，想定する効果を得ることはできなかった．

大臣制度は，自主性が高いチームでは，高い成果を発揮すると予想されるが，そうでないチームでは余計に効率が落ちてしまう．本チームでは，昨年度の体制を踏襲する形で大臣制度を採用したが，効率的ではなかったため，採用する際には，十分に検討を重ねる必要があると考える．

実際に存在した大臣を表 2.1 に示す．

名称	役割
宴会大臣	オンライン飲み会の日程を調整する
ロゴ大臣	ロゴ制作の説明責任を負う
動画大臣	動画制作の説明責任を負う
紹介文大臣	プロジェクト紹介文の説明責任を負う
ポスター大臣	ポスター作成の説明責任を負う
勉強会大臣	勉強会の企画とを調整する
報告書大臣	本報告書の説明責任を負う
提出物大臣	報告書以外の期末提出物の説明責任を負う

表 2.1　各大臣の役割

（文責: 山本竜生）

2.2　ロゴ制作

本プロジェクトでは，プロジェクトの特徴やイメージを表現するとともに，メンバー全員の一体感を生むため，今年度のプロジェクトロゴを制作した．初めに，メンバー全員がロゴ案を 1 つ以上考案した．その後，ロゴ案の発表会を行い，それぞれの案について相互にレビューした．次に，レビューやほかのメンバーのロゴデザインを参考に，各自新たなロゴ案を考案し発表会を行い，各案について再度レビューを実施した．1 回目と 2 回目のレビュー方法は，それぞれの案の良いところだけを伝える形式をとった．その後，再度新たなロゴ案をメンバー全員が考案し，3 回目の発表会を行った．この発表会のレビューでは形式を変更し，良いところだけでなく改善点も指摘した．レビューをすることによって，コミュニケーションをとる機会を設け，良いところだけ伝える形式にすることで発言しやすい雰囲気にできた．さらに，ある程度コミュニケーションをとることができるようになった後で，レビューの形式を変更し改善点の指摘もすることで意見交換をしやすくなった．結果，最初はメンバー間にあったぎこちなさを減らすことができた．その後，レビュー内容をもとにメンバーが投票し，ロゴ案を 2 つに絞り込んだ．ロゴ案を考案したメンバーはレビューをもとに改善案を考案し，4 回目の発表会を行った．この発表会後，メンバーによる投票により，ロゴ案の決定に至った．また，決定したロゴ案の考案者をロゴ大臣とし，ロゴ大臣を補佐するメンバーを 1 名決定した．ロゴ案の考案開始から決定まで 4 週間の時間を要した．その後，さらに 4 週間かけて大臣と補佐がデザイン原案の改善をメンバーや教員の意見をもとに行い，ロゴデザインの最終版を制作した（図 5.4）．このころには，メンバー間の交流も盛んになっており，メンバー間で一体感を生むことができた．

図 2.1　決定したロゴ

図 2.2　その他のロゴ案 1

図 2.3　その他のロゴ案 2

（文責: 大巻佑太）

第 3 章　サービスの考案プロセス

3.1　プロセス概要

本プロジェクトの提供サービス考案までのプロセスは，大きく 4 つに分かれる．1 つ目はロゴの制作である．ロゴの案をプロジェクトメンバー全員が考えて持ち寄り，そこから絞り込みを行っていき，改良を繰り返していった．2 つ目はフィールドワークである．メンバーを 4 グループに分けて，函館やその周辺の街を散策し，新しい発見や課題などを探した．3 つ目は OST（オープンスペーステクノロジー）である．OST を通してアイデアの詳細について議論し，アイデアの詳細について明確にした．4 つ目はブラッシュアップである．OST やアイデアの評価基準をもとにアイデアのブラッシュアップを行った．

（文責: 山口将馬）

3.2　フィールドワーク

3.2.1　事前調査

函館の街やその周辺に存在する問題や課題の調査を目的にフィールドワークを行った．初めに「五稜郭周辺」「函館駅周辺」「木古内周辺」「湯の川周辺」の 4 つの地域で調査することをメンバー間での話し合いと投票で決定した．そして，各地域で実際に調査する 3 人 1 組のグループを編成した．その後，調査するにあたり注意するべきことを確認したうえで各グループタイムスケジュールを組み，それに従いフィールドワークを行った．

（文責: 山内彩土）

3.2.2　フィールドワークに関するレクチャー

フィールドワークの実施にあたって，南部美砂子先生によるフィールドワークについての講義動画を視聴した．動画では，初めにフィールドワークとは何かについての説明があった．フィールドワークとは社会や文化を知るための 1 つの方法であり，得た情報を整理し，出た結論を他人に伝達共有することが大事であると学んだ．次に，フィールドワークは目的ではなく手段の 1 つであり，他者の合理性を理解することが重要であると学んだ．最後に，実例とともにフィールドワークをする調査者のあり方についての説明があった．調査をされる側は迷惑が掛かってしまうことを意識することや，守秘義務と匿名性，調査先で生まれる人間関係に注意して調査をするべきということを学んだ．

（文責: 山内彩土）

3.2.3　プレフィールドワーク

フィールドワークを行う前に，未来大学内でプレフィールドワークを行った．プレフィールドワークはフィールドワークと同じグループで行った．各グループでコロナ禍前と変わったところを探すことを目的とし，フィールドワークへの練習として行い，それぞれ気付いたことを写真や文章にまとめて Slack に投稿し，共有した．その後全体でふりかえりを行った．プレフィールドワークでは 90 分の時間制限を設けて実施したが想定よりも時間が足りなかったという感想が多かったため，フィールドワークのタイムスケジュールの見直しを行った．

（文責: 山内彩土）

3.2.4　フィールドワーク

プレフィールドワークを行った後，3.2.1 節で述べた 4 つの地域，4 つのメンバーに分かれて実施した．なお，本調査では COVID-19 の対策としてなるべく密な状況になることを避け，建物内に入る際の手指の消毒を徹底した．各グループが主に調査した場所は 表 3.1 に記す．調査中には Slack にフィールドワーク用のチャンネルを作成し，写真や文章を共有した．

表 3.1　フィールドワークを行った主な場所

グループ	調査した主な場所
五稜郭周辺	五稜郭公園　シエスタ函館
函館駅周辺	函館駅　朝市　金森倉庫　まちづくりセンター
木古内周辺	函館駅　いさりび鉄道内　木古内駅
湯の川周辺	湯倉神社　市民会館　函館空港

（文責: 山内彩土）

3.2.5　ふりかえり

他グループと共有するため，グループごとにフィールドワークで得た情報をまとめ，フィールドワークで気付いた点をオンラインホワイトボードサービスの Miro を用いて書き出した．その後，全体で結果の報告をした．プレフィールドワークを行った結果，

- 学内の密集を防ぐための張り紙があることで見栄えが悪くなっていること
- 消毒液が置かれている場所の違和感があること
- 視覚に頼るシステムが多いこと
- ライブラリを利用するための整理券が紛失されているものもあったこと
- 各教室の定員人数がわからないこと
- 教室内に何人いるのか把握しにくいため教室内の人数制限が機能していないこと
- トイレのドアが開けっ放しになっていることでどちらがどの性別用のトイレかわかりにくくなっていること

などが分かった.

フィールドワークでは 3.2.1 節で述べた 4 つの地域を調査した結果,

- 函館駅の券売機の場所がわかりにくいこと
- いさりび鉄道内の車両アナウンスが聞き取りづらいこと
- 車両内のコロナ対策がほぼないこと
- 木古内周辺では写真で景色の案内があったこと
- 若い人と年配者での情報を得る方法が違うこと
- 公共交通機関が初めて使う観光客に配慮していないこと
- 観光案内板が景観を損ねてしまうことがあること
- 観光施設の休業や外出している人の減少など COVID-19 の影響が見られること
- 金森倉庫で人力車の仕事をしている人がマスクを着けていなかったこと
- 護国神社までの地図がわかりにくいものになっていたこと

などが分かった.

（文責: 山内彩土）

3.3 サービスの考案

3.3.1 OST(オープンスペーステクノロジー)

アイデアの種に関しての発散，収束した後に，自分が興味関心のあるテーマに沿って深く話し合い，さまざまな観点から考えることができる方法である OST を実施した．フィールドワークで得た興味関心のあるテーマを出し合い，そのテーマごとに議論した．話し合った内容は Miro を用いて，ふせんを使いながら共有した．OST を通じて，興味関心の偏りを可視化し，アイデアの種がより具体化された.

（文責: 林杏実）

3.3.2 アイデアの評価基準を決定

アイデアを選定するために，1.3 節で述べたプロジェクトの目的に沿った評価基準が必要となった．その評価基準をメンバーで話し合った結果，以下の 4 点を評価基準と定めた.

- 「ビーコンである理由」: ビーコンを用いてサービスを提供する点から，ビーコンならではのサービスであるのか評価する.
- 「函館らしいサービス」: 函館の街なかにビーコンを設置する点から，函館ならではのサービスであるのか評価する.
- 「サービスの新規性」: 利用者に新しい価値を創造するサービスを提供する点から，考案されたサービスに新しい価値が含まれているのか評価する.
- 「サービスのおもしろさ」: 開発したサービスに興味を持ち，利用率を上げる点から，考案されたサービスはユーザーがおもしろいと感じるかを評価する.

（文責: 林杏実）

3.3.3　アイデアの絞り込み

50 を超えるアイデアに対し，グルーピングを行い，再度アイデアについて OST を用いてブラッシュアップした．そのアイデアについて熟考し，必要な価値であるかの取捨選択を Miro を用いて行った．また，OST を重ねた上で，メンバーが推薦するアイデア選出し，アイデアを絞り込んだ．今後は，そのアイデアの中から 3.3.2 節で述べた評価基準を用いて，最終的なアイデアを選定していく．

（文責: 林杏実）

3.3.4　テーマの決定

ブラッシュアップの結果得られた 6 つのアイデアについて，Google Form を用いて評価をした．3.3.2 節で述べた評価基準を用いて，4 段階で評価した．

表 3.2　アイデア評価一覧表

	函館らしさ	ビーコンらしさ	新規性	おもしろさ	合計	順位
未来大交流パーク	2.50	3.42	2.92	3.17	11.32	3
朝市水族館	4.00	2.42	3.33	3.08	12.35	1
ジュークボックス	1.33	3.25	3.25	3.58	10.77	4
バイク	1.33	2.33	3.50	2.75	9.45	6
椅子	1.33	2.75	3.75	2.67	9.95	5
動物なりきりシステム	2.33	3.08	3.42	3.50	11.72	2

上の表は，小数点第 3 位で四捨五入して結果を示している．結果，朝市水族館，動物なりきりシステム，未来大交流パークが上位となった．しかし，上位なったアイデアはメンバーがサービスとして開発したいと思っていたアイデアとは異なり，プロジェクトメンバーが以上の結果に納得できなかったため，学内サービス「LATTE」の改修と，過去のビーコンプロジェクトのサービス「Telepath」の改修を加えた 8 つのサービスで人気投票をした．人気投票において妥当性はなく，評価基準よりもプロジェクトメンバーがこのアイデアをもとにサービスを開発したいかどうかを優先し，メンバーの合意のもと人気投票をした．結果，ジュークボックス，LATTE，朝市水族館が上位となった．評価結果を照らし合わせて LATTE の改修と動物なりきりシステムを学内システムとして統合し，朝市水族館，ジュークボックス，学内システムに決定した．3 つのアイデアの中から各自が開発したいと考えるサービスを選び，話し合いによってグループメンバーを決定した．

（文責: 小田嶋亜美）

第 4 章　開発

4.1　アジャイル開発とスクラムの概要

本プロジェクトでは，開発手法としてソフトウェア開発手法の 1 つであるアジャイル開発を導入した．アジャイル開発とは，プロセスやツールよりも個人と対話を，包括的なドキュメントよりも動くソフトウェアを，契約交渉よりも顧客との協調を，計画に従うことよりも変化への対応を価値とする開発手法である [6]．アジャイル開発では，小さな単位で実装とテストを繰り返して開発をするため仕様変更に強く，動くソフトウェアを早く継続的に提供できる．また，私たちは 2021 年 6 月 2 日にオンラインで開催されたスクラムワークショップ 2021 に参加し，アジャイルおよびスクラムについて学んだ．その後，ウォーターフォールやプロトタイピングなど，そのほかの開発手法についてもプロジェクトメンバーで勉強会を開催した．その結果，短い期間で価値のあるソフトウェアを開発するにはスクラムを採用することが最適だと考え，全チームがアジャイル開発手法の 1 つであるスクラムを採用することとなった．

スクラムについて，スクラムガイドでは「スクラムとは，複雑な問題に対応する適応型のソリューションを通じて，人々，チーム，組織が価値を生み出すための軽量級フレームワークである．」と定義されている [7]．スクラムは，「経験主義」と，無駄を省き本質に集中する「リーン思考」にもとづいており，「透明性」「検査」「適応」の三本柱を実現することで機能する．

スクラムでは，スクラムチームという小さなチームを構成する．スクラムチームは，プロダクトオーナー 1 人，スクラムマスター 1 人，開発メンバー複数人によって構成される．通常は 10 人以下で構成され，それぞれのロールには上下関係が存在せず，明確な責任が存在する．プロダクトオーナーは，プロダクトの価値を最大化することの結果に責任を持つ．主にスクラムチームの長期的な目標であるプロダクトゴールの策定や，ユーザーに体験させたい機能や価値に優先順位をつけてリスト化したプロダクトバックログを管理する役割がある．それらの作業はほかの人に委任しても良いが，最終的な責任はプロダクトオーナーが持つ．スクラムマスターは，スクラムを正しく実践できるように支援し，スクラムを円滑に進行することに責任を持つ．スクラムマスターはさまざまな形でスクラムチームやプロダクトオーナーに対して支援をする全体の奉仕者であり，真のリーダーのような役割である．開発メンバーはスクラムチームの一員であり，4.2 節で後述するスプリントにおいて，品質の確かなプロダクトを作成することに責任を持つ．

（文責: 柿本翔大）

4.2　スクラム開発のながれ

スクラムは，スプリントと呼ばれる最長 1 ヵ月の短い開発期間を繰り返すことで進められる．スプリントの期間内には開発作業のほかに，スプリントプランニング，デイリースクラム，スプリントレビュー，スプリントレトロスペクティブという，プロダクトゴールを達成するための 4 つのイベントが行われる．

スプリントプランニングは，スプリントで実行する作業の計画をスクラムチーム全体で立てるイベントであり，スプリントの最初に行われる．スプリントプランニングでは，まずそのスプリントになぜ価値があるか話し合い，スプリントの目標であるスプリントゴールを定義する．次に，そのスプリントで何ができるのか話し合い，そのスプリントで実施するプロダクトバックログアイテムを選択する．最後に，選択した作業をどのように成し遂げるのか話し合い，スプリントバックログを作成する．スプリントバックログとは，スプリントゴールと選択したプロダクトバックログアイテムに加え，その価値を提供するために作成した計画の総称である．

デイリースクラムは，スプリント期間中毎日，同じ時間・同じ場所で行われる，開発メンバーのための 15 分のイベントである．スプリントゴールに対する進捗を検査し，必要に応じてスプリントバックログを調整することを目的として行われる．デイリースクラムでは，スプリントゴールの進捗に焦点を当て，次のデイリースクラムまでに行う作業の計画について話し合う．そうすることで，コミュニケーションの改善，障害物の特定，迅速な意思決定を促進し，ほかの会議を不要とする狙いがある．

スプリントレビューは，スクラムチームがステークホルダーに対してスプリントの成果を示し，プロダクトゴールに対する進捗について話し合うイベントである．ステークホルダーから得た意見をもとにプロダクトバックログの見直しなどをすることを目的として行われる．本プロジェクトでは，スプリントレビューの時間がプロジェクト学習の講義時間に収まるよう，各チーム 30 分の持ち時間で発表，デモンストレーション，質疑応答を行い，これをスプリントレビューとした．

スプリントレトロスペクティブは，スクラムチームでスプリント期間中の行動や出来事を振り返るイベントである．スプリントでうまくいったことや，いかなかったこと，およびそれらの理由を考え，話し合うことで次回以降のスプリントを改善することを目的として行われる．このイベントをもってスプリントを終了する．

（文責: 柿本翔大）

4.3 スクラム開発の効果と課題

4.3.1 効果

スプリント期間中にデイリースクラムを行い，チーム内で困っていることを共有することで問題を早期に解決できた．スプリントの終了時には，プロジェクト全体でスプリントレビューを行った．ほかのチームや先生方から意見をもらい問題点を発見できた．短いスプリントで開発とレビューを繰り返すことで，新たな課題を見つけやすく，修正しやすい効果があった．また，必要な機能を優先的に開発することにもつながった．スプリントレビューの後はスプリントレトロスペクティブを行った．スプリント期間中の良かったところと悪かったところをチーム内で話し合い，早期の問題発見や作業の効率化などの効果が見られた．

（文責: 佐々木紀明）

4.3.2 課題

スクラム開発を採用することによって前述したような効果が出たが，課題も見つかった．デイリースクラムでは集合時間に起きることができなかったり，別の予定が入ってしまったりしてメンバーがそろわないことも何度かあった．また，進捗を報告するだけの場になってしまうこともあり，問題の特定などができず有意義な時間とは言えないこともあった．スプリントレビューでは準備が不足し，機能の必要性やサービスの魅力をうまく伝えられないことがあった．スプリントレトロスペクティブでは問題の有効的な解決策が浮かばず，何度も同じ問題を挙げることがあった．スクラム開発に慣れていないため，開発の終盤にタスクが集中してしまった．

（文責: 佐々木紀明）

4.4 開発したサービスについて

4.4.1 あさいち水族館

函館朝市内のお店の情報や魚の情報を閲覧できるサービス「あさいち水族館」を開発した．あさいち水族館は，地元の方々に函館朝市をもっと楽しんでもらいたい，もっと利用してもらいたいという思いが込められて開発されたサービスである．あさいち水族館を開発した理由は，函館朝市に訪れた際，現場の方から地元客の利用が少ないという意見があった．そこで函館にはない新しいものを考案し，「函館朝市」と「水族館」をかけ合わせサービスの開発に至った．本サービスの詳細については第Ⅱ部に記す.

（文責: 石田海祐）

4.4.2 Favor

黙食などで会話の制限された場所での重たい雰囲気を緩和するための音楽共有サービス「Favor」を開発した．本サービスは，同じビーコン範囲内にいる利用者どうしで Spotify の音楽を共有できる．友人どうしだけでなく知らない人との共有も可能になっており，音楽を通じた一期一会のコミュニケーションを可能にしている．開発は山内，佐々木，萩原の 3 名で行い，アジャイル開発手法の 1 つであるスクラムに挑戦した．本サービスの詳細については第Ⅲ部に記す.

（文責: 山内彩土）

4.4.3 FLAT

チーム FUN Location は，未来大内で友だちの居場所を連絡をとらずに訪ねるのは難しいという問題を解決するために，未来大内の位置情報を共有するサービス「FLAT」を開発した．本サービスは，未来大内に設置されたビーコンを用いて未来大内の友だちの居場所を表示でき，気軽に友だちのもとへ会いに行ける．本サービスの詳細については第Ⅳ部に記す.

（文責: 小田嶋亜美）

第 5 章　成果発表会

5.1　中間発表会

5.1.1　発表形式

中間発表会はオンラインで開催された．本プロジェクトの概要や，サービスを企画する際に用いた仮アイデアを説明するため，10 分間の動画とポスターを使用して発表した．その後，Zoom を利用して 15 分間の質疑応答を 6 回行った．

表 5.1.1 中間発表スケジュール

15:00 ~	プロジェクト報告公開開始
16:00 ~16:15	前半質疑応答 1
16:20 ~16:35	前半質疑応答 2
16:40 ~16:55	前半質疑応答 3
16:55 ~17:05	休憩
17:05 ~17:20	後半質疑応答 1
17:25 ~17:40	後半質疑応答 2
17:45 ~18:00	後半質疑応答 3

5.1.2　レビュー内容

発表形式の評価と反省

発表形式に関して得た高評価な意見は

- 「字幕を入れていて動画を見ていて内容が頭に入りやすかった。」
- 「スライドが見やすく分かりやすかった。」
- 「ポスターとてもレイアウトもきれいで無駄のない印象をうけた。」
- 「丁寧かつゆっくりと質疑応答に返答して聞きやすかった。」

などが得られた．アンケートは 39 件の回答が得られ平均評価は 10 点中，7.4 点であった．以上から発表形式に関しては概ね高評価を得ていることが分かった．それに対し，低評価な意見として

- 「ビデオの読み上げが単調であったように感じた。」
- 「動画のほうはもう少し実際に行ったものの具体的な説明がほしいなと思った。」
- 「明確な回答をできてないところがあったからそこを気を付けた方がよい。」
- 「発表者の後ろに大人数いるのか、すごくノイズが大きくて気になった。」
- 「スライドに乗っていない情報も発表に組み込んだ方がよい。」
- 「質問が出ないとき静かで進んでいなかった。」

などが得られた．反省すべき点として，動画の読み上げが単調なことや実際の活動について具体

的な説明が少ないこと，質問に対して明確な回答ができていないことが挙げられた．またスライドにある情報しか発表していないことや質問がない時に無言の時間ができてしまったこともあった．これらの意見を踏まえて，ふりかえりをした結果

- 「動画やポスターの内容が不十分であったので早めに作成し、レビューは早めにするべきだ。」
- 「注目させたい情報を発表する際には抑揚をつける。」
- 「動画での発表を複数人でするべきだ。」
- 「発表用の資料を作成するべきだ。」
- 「発表練習をするべきだ。」
- 「質疑応答の際に質問が出ない場合はポスターやプロジェクトなどの説明をするべきだ。」
- 「スライドにない情報も発表するべきだ。」

などの意見が得られた．以上から，動画やポスターのレビューを早めに行うため，余裕を持ってそれらを作成すること．発表用の資料を作成し事前に発表練習をする必要があるとわかった．今回の反省を次回の発表会に活かしていきたい．

（文責: 中山寿似也）

5.1.3 発表内容の評価と反省

発表内容に関しては，高評価な意見として，

- 「どういった方法でプロジェクトの進行をしたかの説明も多い、活動が把握しやすい内容だった。」
- 「アイデアがコンセプトやターゲットなどが明記されていてわかりやすかった。」
- 「大臣制度を取ることで全員がさまざまな役職を担当できることがよいなと感じた。」

などが得られた．低評価な意見としては，

- 「プロジェクトの実際の活動の部分が薄いなと感じた。」
- 「大雑把なアイデアで、詳細までつめられていないと感じた。」
- 「実装しようと考えている内容が本当にビーコンを使う必要があるのか、その内容で目標を達成できるのかがよく練られていないように感じた。」

などが得られた．アンケートは 39 件の回答が得られ平均評価は 10 点中，7.4 点だった．プロジェクトの進行やほかのプロジェクトにはない大臣制度に関して高評価な意見が多かった．しかし，活動内容の目的や目標が明確ではなかった部分や，仮アイデアの内容が大雑把であり伝わらない部分もあった．また，仮アイデアの内容に対してグループ内での理解度の差が生まれてしまった．今後は発表会に対しての見通しとスケジュールをしっかり立て，準備していきたい．

（文責: 小田嶋亜美）

5.2 成果発表会

5.2.1 発表形式

2021 年 12 月 10 日，オンラインで行われたプロジェクト学習成果発表会で発表した．発表形式はサービスを作成した動画を見てもらい．質疑応答するというものだった．また，質疑応答の時間に，簡単にサービスの概要説明を繰り返し行った．各グループの背景，サービスの概要，ユーザーストーリーなどを説明した動画とポスターを作成して発表に用いた．

（文責: 石田海祐）

5.2.2 レビュー内容

発表形式の評価と反省

発表形式に関して得られた高評価な意見は，

- 「実際に担当しているメンバーが動画内で出演して説明しているという点が、非常に良いと思った。オンラインで行っている以上、実際に人が説明しているところはなかなか見られないので、良い方式だと思った。」
- 「製作物の簡単な説明から始まり実際に使っているところを見せてくれたので、とても分かりやすくできていると思った。」
- 「動画は字幕もあり、声も聞き取りやすく、とても分かりやすかった。」
- 「3 つの成果物を丁寧には説明している印象が良かったです。」
- 「話すスピードが丁度よく聞き取りやすかった。」

など多くの肯定的な意見が得られた．アンケートの回答は 40 件寄せられ，評価の平均得点は 10 点満点中 8.7 点であった．このことから，発表形式は前期の反省点を改善して良い発表にできたことがわかる．それに対して，否定的な意見として

- 「質問するタイミングが難しいと感じたので、概要をサッと説明するだけでも良かったのかなと思いました。」
- 「質問が来た時に流れをぶった斬る形になっていたので、概要だけ先に全て説明して良いのでは無いかと思います。また、評価フォームへの導線があるといいですね。」

と，質疑応答の沈黙をなくそうとしたがために，逆に質問者が質問するタイミングがないと感じさせてしまっていた．これに関しては書いていただいた意見通り，先に概要だけを説明すべきだった．

（文責: 石田海祐）

発表内容の評価と反省

発表技術に関して得られた高評価な意見は，

- 「使っている様子なども伝わる発表内容でした。」

- 「製作物が形となっていため、内容により理解しやすさがあった。また、製作物の UI がどれも見やすくできていていいと思った。」
- 「発表もわかりやすかったです、そして本アプリを使いたいなと思いました。」
- 「もともとあるアプリとの比較点を明示しているのはとてもいいと思った。」
- 「とても明確化されていた。」

などと，発表技術に関しての評価でも肯定的な意見は多く得られた．評価の平均得点は 10 点満点中 8.5 点だった．このことから，発表技術に関しても評価は高く，前期の反省が活かされ，点数が上がったと考えられる．これに対して，否定的な意見は，

- 「想定する完成形へ向けて必要な技術や知識を様々なものを取り入れられている点が優れていた。その分目標を達成できていなかったのが残念。これからの進捗に期待したい。」
- 「Presentation. Your video presentation is much clearer. I like the way you have included your development process. The recorded examples 'in action' are great because they exemplify exactly what your technology does. Some good video editing too. The restaurant example was good but the most innovative example in my opinion was the apartment location. Good work but I suggest your poster is not a informative as your excellent presentation. プレゼンテーション。ビデオによるプレゼンテーションは、より分かりやすくなっています。あなたの開発プロセスを含めた方法が気に入っています。記録された「実際の」例は、あなたの技術が何をするのかを正確に例示していて素晴らしいです。ビデオの編集も良いですね。レストランの例も良かったですが、私の意見では最も革新的な例はアパートのロケーションでした。良い作品ですが、ポスターはあなたの優れたプレゼンテーションに比べて情報量が少ないと思います。」

と，ポスターの内容の薄さや目標を達成できなかったことに関してなどの否定的なコメントが見られた．ポスター内容はどうがに比べて省略されている部分があったため，省略せず書くべきであった．

（文責: 石田海祐）

第 6 章　まとめ

本プロジェクトでは，ビーコンを用いて函館のまちのさまざまな魅力や新たな価値を生み出すことを目的として，アジャイル開発手法を用いて「あさいち水族館」「Favor」「FLAT」の 3 つのサービスを提案，開発した．これらのサービスを開発したことにより，函館のまちの価値，魅力を向上することに貢献できたと考える．また，1 年間のプロジェクト学習を通してアジャイル開発手法の学び，サービスの立案，ファシリテーター制度など，自分たちで物事を考えて活動を進められた．しかしながら，中間発表，成果発表の時のポスターや動画の準備は，後半に詰め込んだ形になってしまったためもっと早めに準備をしておくべきだったと．また，報告書や成果物のレビューは先生方や TA の方からはしてもらえたものの，学外の方，またはそれ以外の方からのレビューをいただく時間がなかったので，今後に外部発表があるサービスもあるため，今回の反省点を改善して外部発表に挑みたい．

（文責: 石田海祐）

参考文献

[1] **総務省｜令和 *2* 年版 情報通信白書｜企業における *IoT*・*AI* 等のシステム・サービスの導入・利用状況**. https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r02/html/nd252150.html. (Accessed on 07/19/2021). 2020.

[2] **総務省｜令和 *2* 年版 情報通信白書｜情報通信機器の保有状況**. https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r02/html/nd252110.html. (Accessed on 07/21/2021).

[3] *Bluetooth Technology Overview | Bluetooth® Technology Website*. https://www.bluetooth.com/learn-about-bluetooth/tech-overview/. (Accessed on 07/19/2021).

[4] *Coke ON* **（コーク オン）- おトクで楽しいコカ・コーラ公式アプリ**. https://c.cocacola.co.jp/app/. (Accessed on 07/19/2021).

[5] *Getting Started with iBeacon*. URL: https://developer.apple.com/ibeacon/Getting-Started-with-iBeacon.pdf (visited on 01/18/2022).

[6] **アジャイルソフトウェア開発宣言**. https://agilemanifesto.org/iso/ja/manifesto.html. (Accessed on 01/07/2022).

[7] *2020-Scrum-Guide-Japanese.pdf*. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Japanese.pdf. (Accessed on 01/07/2022).

第 II 部

あさいち水族館について

第 1 章　背景

1.1　背景と課題

函館は，全国的にも知名度が高い観光地であり，観光は函館の主要な産業の 1 つである．函館市の観光入込客数は札幌市の 571 万人についで 2 位の 310 万人である [8]．また，海に囲まれた街であり海産物で有名である．その中で函館朝市は有名な観光スポットの 1 つである．函館朝市では新鮮な海鮮物を食べることやお土産を買う事ができ，観光客からも人気である．しかしながら，近年は COVID-19 の影響で全国的に自粛ムードが続き旅行する機会が減り，観光客の減少が続いている．観光庁の調査 [9] によると，日本人国内旅行消費額は 2019 年 7-9 月の消費額は 66932 億円であったが，2020 年の同じ時期の消費額は 29028 億円と，COVID-19 が蔓延してから消費額は半分以下になっている．「自粛ムード」というものが収まれば良いが，感染症の蔓延がいつ収まるかなどは誰もに分からない．もしこのまま感染症の蔓延が収まることがなければ，函館の街は大打撃を受けてしまうことは避けることができない．観光地というものは地元の人が足を運ぶ回数がどうしても下がってしまう．特に函館朝市などは観光客で賑わっていることが多い．函館市の調査 [10] によると，函館の観光客のうち 68.1% が函館朝市に訪れた．もしくは訪れる予定という結果が出ている．そこで，私たちは函館朝市を盛り上げるために地元の人をターゲットにした．地元の人に定期的に来てもらえるようなサービスを提供ができれば，函館朝市のお客さんを増やすことができ，函館朝市を盛り上げる事ができると考える．また，函館には水族館がなく函館に住んでいる子どもたちは水族館になかなか行くことができない．そこで，私たちは函館朝市と水族館を組み合わせてみるという考えに至った．これにより，函館朝市内で水族館を擬似体験できるようなサービスを提供できれば，子ども連れの家族をターゲットにして函館朝市で集客効果を狙う事ができ，同様に函館朝市を盛り上げることができるのでないかと考える．

（文責: 山口将馬）

第 2 章　目的

2.1　目的

本グループには 2 つの目的がある．1 つ目は本サービスを通して観光客だけでなく，地元の人にも函館朝市に足を運んでもらいを函館朝市を盛り上げることである．2 つ目は函館にはない水族館をアプリケーションを通して擬似体験してもらうことである．また，函館朝市に水族館要素を取り入れることで子ども連れの家族の集客を見込む事ができる．多くの集客により，函館朝市を盛り上げることで函館の街全体を盛り上げることができると考える．

（文責: 山口将馬）

第 3 章　サービスの概要

3.1　サービスの概要

「あさいち水族館」は，函館朝市に訪れた方が函館朝市を散策しながら自分のスマートフォンで魚やお店の情報を閲覧し，自分だけの水族館を完成させるサービスである．このサービスでは，ユーザーに函館朝市内のお店の情報や魚の情報を知ってもらうと同時に，函館朝市で見つけた魚を自分の水族館にコレクションする機能を提供する．

本サービスを利用することで，函館にはない水族館を函館朝市で体験でき，函館朝市をより楽しむことができる．それにより函館朝市に地元の方がたくさん訪れるようになり，函館朝市の活性化にもつなげることができる．

（文責: 中山寿似也）

3.2　ユーザーストーリー

本サービスは，サービスを提供する場所を函館朝市とし，想定するターゲットを函館朝市に訪れた人としている．ここでは函館で暮らす大学生を想定したユーザーストーリーを述べる．ユーザーはアプリケーションをインストールする．ユーザーはガイド外面でアプリケーションの使い方を見る．ユーザーは函館朝市に訪れ，アプリケーションを開きながら函館朝市内を歩く．函館朝市内のお店の近くにいくと，アプリケーションの画面内に魚が飛び込んでくる．その魚をタップするとその魚がどのような魚か知ることができる．そこでお店の情報も知ることができる．ユーザーはアプリケーションの画面内に飛び込んできた魚の中から好きな魚を自分の水族館にコレクションして楽しめる．この体験をもう一度したいとユーザーが思い，また函館朝市に行きたいと感じる．以上があさいち水族館のユーザーストーリーである．

（文責: 中山寿似也）

第 4 章　開発

4.1　アプリケーションの機能

本アプリケーションでは，主に 3 つの機能を実装した．「魚を探す機能」，「自分の水族館機能」，「使い方ガイド機能」である．

（文責: 田村修吾）

4.1.1　魚を探す機能

ビーコンを検知すると魚のイラストが出現する．出現した魚をタップすると，対応する魚の情報を見ることができる．さらにこの魚がいるお店ボタンをタップすると，対応するお店の情報も見ることができる．

（文責: 田村修吾）

4.1.2　自分の水族館機能

ビーコンを検知して見つけたお魚を自分の水族館に追加し，一覧で見ることができる機能．一覧表示された魚をタップすることで，詳細な魚の情報を見ることができる．

（文責: 田村修吾）

4.1.3　使い方ガイド機能

本アプリケーションの使用方法について知ることができる機能．「魚を探す機能」，「魚やお店の情報閲覧機能」，「自分の水族館機能」の 3 つの機能の使い方を説明する．

（文責: 田村修吾）

4.2　利用したツール・サービス

4.2.1　Git

ソースコードのバージョン管理をするときに利用した．Git は分散型バージョン管理システムである．ローカルリポジトリに各チームメンバーの作業を保存し，作業内容を共有するときにリモートリポジトリへアップロードする．

（文責: 田村修吾）

4.2.2 Slack

各プロジェクトメンバーや教員，TA の方々との連絡ツールとして Slack を用いた．チームコミュニケーションツールである Slack では，アンケートやファイルの共有を容易に行うことができた．

（文責: 田村修吾）

4.2.3 Discod

チームメンバーとの会議に，ビデオ通話，音声通話サービスとして Discod を用いた．通話への参加や作業画面の共有が容易であるため，円滑な会議ができた．

（文責: 田村修吾）

4.2.4 LINE

チームメンバーとの連絡に，音声通話，メッセンジャーサービスとして LINE を用いた．特定の人物のみ閲覧，投稿可能である，グループ機能を利用することで，チームメンバーとの連絡，共有ツールとして利用した．

（文責: 田村修吾）

4.2.5 Zoom

プロジェクトメンバーや教員，TA の方々との会議に，Web 会議サービスである Zoom を用いた．週 2 回行われるシステム情報科学実習の時間では，2 週に 1 回のみ対面での活動が可能だったため，リモート活動行う上で必須のサービスだった．ブレイクアウトルームと呼ばれる Web 会議のチャンネルを複数作成する機能を用いて，プロジェクト活動を円滑に進めることができた．

（文責: 田村修吾）

4.2.6 Miro

プロジェクトメンバーとの会議に，オンラインホワイトボードツールとして Miro を用いた．リモート活動でメンバー全員での同時編集ができたため，円滑な意見交換や会議ができた．

（文責: 田村修吾）

4.2.7 esa

議事録の作成や管理に，情報共有サービスの esa を用いた．複数人での同時編集ができ，マークダウン記法で資料を作成できる．esa には，マークダウン記法を利用したことがない人でも使いや

すいように，さまざまな入力補助がある．そのため，マークダウン記法を利用したことがないプロジェクトメンバーでも容易に利用できた．

（文責: 田村修吾）

4.2.8 Figma

サービスデザインの制作に，Web デザインツールである Figma を用いた．画面設計，動線設計，デザインを容易に確認できた．

（文責: 田村修吾）

4.2.9 Canva

画面デザインやロゴの制作に，デザイン作成ツールの Canva を用いた．

（文責: 田村修吾）

4.2.10 GitHub

サービスメンバー間でのコード共有に，Git を利用したソースコードのバージョン管理サービスである GitHub を用いた．複数人でのチーム開発を円滑に進めることができた．

（文責: 田村修吾）

4.3 技術習得

本サービスの開発メンバーは全員が開発未経験だったため，大まかなプラットフォーム分けと使用する言語の選定をした．その中でサービスを開発する上で必要な技術の学習をした．メンバー個人で自分の担当する箇所を学習して，そこで学んだことをメンバーに共有した．メンバー全員の開発経験がないため発言しにくい空気になることが予想された．それを回避するため，自由な発言をしやすように Zoom で学んだことの資料を共有しながら雑談形式で行った．
しかし夏休み中に開催したということもあり，メンバー全員で参加できないことが多かった．その際にはメンバーが作成した資料を自分で学習し，わからないことがあったら質問するという形式にした．これらの活動を通して GitHub の基本的な使い方や知識，Swift 言語や PHP 言語のコーディング方法といった各プラットフォームの知識を得ることができた．これにより，開発を円滑に行うことができた．

（文責: 中山寿似也）

4.4 導入した開発手法

4.4.1 アジャイル開発とスクラムの概要

本プロジェクトではアジャイル開発手法の 1 つであるスクラムを採用した．スクラムを導入するにあたり，スクラムチームを形成した．スクラムチームの構成としてはプロダクトオーナー兼開発チーム 1 人，スクラムマスター兼開発チーム 1 人，開発チームが 4 人である．

本サービスは iOS アプリケーションを作成することに決定した．開発チームはデザイン，ビーコン検知，サーバーサイドアプリケーションに分かれた．本グループは，プロダクトバックログ，スプリントバックログを Miro で管理した．プロダクトバックログを作成する際にはまずユーザーストーリーマップを作成し，ユーザーストーリーを洗い出した．そこからユーザーストーリーに優先順位をつけ，それぞれの作業量をメンバーで話し合いながらプロダクトバックログを作成した．そこからスプリントバックログを作成し，スプリント期間を話し合い最初の第 1 スプリントを 2 週間と定め，以降のスプリントから 1 週間ごとの期間とすることにした．プロジェクト全体での話し合いでスプリントの終了日を，毎週水曜のプロジェクト活動時間とした．スプリント開始時にスプリントプランニングを行い，毎週水曜日のプロジェクト活動時間にスプリントレビュー，スプリントレトロスペクティブを行ってスプリントを終えた．しかし，最初のスプリントで開発期間が間に合わないと判断し，第 1 スプリントを 3 週間に変更することになってしまったことやスプリントゴールを達成できていないことがあった．これらの要因として，スプリントバックログを作成した際に適切な作業量の見積もりができていなかったこと，開発する機能を途中で変更したことがある．

スプリント期間中は，LINE 上で毎日 21 時から 20 分間のデイリースクラムを実施した．デイリースクラムでは，メンバー全員で「今日やったこと」「明日やること」「困っていること」を共有し，スプリントゴールが達成できそうか，どのように問題を解決するかを話し合った．その際に，全員が必ず毎回参加できるわけではなく，参加していない人がどこまで作業できているのかがわからないことがあった．そこでデイリースクラムに参加できない人は esa に「今日やったこと」「明日やること」「困っていること」を書いてメンバーに共有することで，メンバー全員の作業を透明化できた．スプリントプランニングは，バックログの項目から今回のスプリントで完成させたい機能の決定をした．そのために必要な作業とその時間の見積もり，スプリントバックロゴの作成をした．

スプリント中の開発では GitHub の PullRequest やメンバーどうしで画面共有をしながらプログラミングを行った．スプリントレビューはプロジェクト全体でプロダクトの進捗報告とスプリントで実装した機能のデモを行った．他のメンバーからのレビューを受けることでプロダクトの核となる機能や，機能の改善をした．スプリントレトロスペクティブはメンバー全員で「よかったこと」「問題点」「改善方法」について振り返った．

（文責: 中山寿似也）

4.5　開発の流れ

4.5.1　スクラム導入の効果と課題

効果

「4.4.1 アジャイル開発とスクラムの概要」で述べた開発手法を取り入れたことによりいくつかの効果が得られた．スプリント期間ごとに実装する機能を決めていくのでサービスの急な変更に対応ができた．デイリースクラムを行ったことにより，メンバーの進捗状況や困っていることなどを共有することで，メンバー個人の進捗状況のにつながり，解決すべき問題にすばやく対応できた．スプリントバックログの作成により，チームメンバーが各自何をすれば良いのかが細分化され，作業を効率化できた．そのほかにも，メンバー各自が何の作業まで終わらせれば良いのか明確になり，役割を正確に把握できたまた，コロナの影響でオンラインでの活動が多い中，チームメンバーとのコミュニケーションを取ることで仲を深め，メンバーが発言しやすい雰囲気を作れた．スプリントという単位で開発期間を区切ることで実装する機能が細分化し，開発しやすくなった．スプリントの最後にスプリントレビューにより，自分たちが気付くことのできなかった問題点をフィードバックとして得ることで，サービスの見直しや改善につながった．その際に，スプリントレビューまでに完成させるという締め切り意識にもつながり，チームメンバー全員で作業を効率化できた．スプリントレトロスペクティブを行うことで，開発についての良かったこと悪かったところをふりかえり改善点を見つけることができた．全体を通してメンバーとコミュニケーションを取る機会が多いため，メンバー間で情報の共有がしやすく共通認識を持つことができた．

（文責: 中山寿似也）

課題

スクラムを導入したことによりいくつかの課題が見つかった．まず，デイリースクラムは毎日決まった時間にメンバー全員で行うが，用事や，作業などでメンバー全員が参加できていない日があった．その影響でメンバーの進捗状況を確認できずに，スプリントゴールを達成できないことがあった．デイリースクラムの重要性をメンバー全員が理解しておらず，必ず参加するという意思がなかった．この問題の解決策として，参加できないメンバーには esa にデイリースクラムの内容を投稿してもらうことにした．またデイリースクラムが進捗報告だけにならないように，スプリントゴールを意識してそれが達成できそうか，スプリントバックログを確認しながら話し合うことにした．これによりスプリントゴール達成までの計画を再検討しやすくなった．また，スクラムは変更に対応しやすいという利点があるが，変更できてしまうことによって開発期間が伸びるという課題があった．チームメンバーの開発経験がないことを踏まえれば，最初に決めたアイデアを変更せずに最後まで開発する手法の方がチームに合っていた．しかし，スクラム開発に挑戦してみたいというチームメンバーの意思でスクラム開発をすることにしたため，技量や知識が不足していたことが原因である．チームメンバー全員がスクラム開発，開発そのものについての経験値のない状態でスクラム導入は，難しいことがわかった．

（文責: 中山寿似也）

第 5 章　実装

5.1　システム構成

「あさいち水族館」はビーコン，モバイルアプリケーション，サーバーサイドアプリケーションの 3 つから構成されている．ビーコンは，特定の UUID，Major 値，Minor 値，を有した発信機としての役割を担い，スマートフォンがビーコンの発する特定の電波を検知する．今回は iBeacon という規格を用いる．本サービスでは函館朝市内のお店ごとにビーコンを設置し，函館朝市内を散策した際にモバイルアプリケーションで電波を受信することで，魚やお店などの情報を取得する．モバイルアプリケーションでビーコン電波の検出，そのビーコンに対応した魚やお店の情報を表示する．サーバーサイドアプリケーションでは，モバイルアプリケーションに魚やお店などのデータを送信する．

（文責: 中山寿似也）

5.2　モバイルアプリケーション (iOS)

本サービスは，モバイルアプリケーションとして iOS の環境で動作するように開発をした．チームメンバー全員に開発経験がないため，iOS と Android を両環境での開発は難しいと判断し，普段から利用している iOS 環境のモバイルアプリケーションを開発した．

（文責: 中山寿似也）

5.2.1　開発環境

開発ツールとして Xcode13.1 を使用し，開発言語は Swift5.5.1 を用いた．ライブラリ管理ツールとしては CocoaPods を使用し，バージョンは 1.10.1 であった．画面設計のフレームワークとしては Storyboard を使用した．またアプリケーション内でアニメーションを導入する際にアニメーションライブラリの Lottie を使用した．

画面設計に Storeboard を使用した理由として，メンバー全員に開発経験がないため GUI 操作で直感的にビューとコードを結ぶことができる Storeboard を用いて開発をした．

（文責: 中山寿似也）

5.2.2　各画面の説明

ホーム画面

アプリケーションを起動したときに表示される画面である．図 5.1 のように，画面の上部には，あさいち水族館のロゴが表示され，画面の下部には，各機能の画面への遷移ボタンが表示される．

各機能とは，前述の通り「魚を探す機能」，「自分の水族館機能」，「使い方ガイド機能」の 3 つである．

（文責: 田村修吾）

図 5.1　ホーム画面

ビーコン検知画面

ホーム画面から，魚を探すボタンをタップすると，ビーコン検知画面へと遷移する．ビーコンを検知していない場合，背景画像，戻るボタン，画面名のみが表示される．ビーコンを検知すると，図 5.2 のように，検知したビーコンに紐づけられた魚のイラストが表示される．表示された魚をタップすると，対応する魚の情報表示画面へと遷移する．

（文責: 田村修吾）

図 5.2　ビーコン検知画面

魚の情報表示画面

魚の名前，魚のイラスト，魚の生息地，詳細な説明を表示する．魚のイラストはフリーイラストサイトである，「illust image」，「イラスト AC」，「素材のプチッチ」，「いらすとや」から使用した．本物の魚の画像ではなく，イラストを使用した理由は，想定するユーザーが主に函館朝市を利用する親子であることから，ある程度カジュアルな魚のイラストを使用したほうが良いと判断したからである．検知したビーコンに紐づけられた魚の情報を表示させるため，UIKit の UIScrollVeiw を利用した．スクロール表示を可能にすることで，限られた範囲内に多くの文章を表示させることができる．図 5.3 のように，画面の下部にこの魚がいるお店ボタンと，自分の水族館に追加ボタンがある．この魚がいるボタンをタップすると，対応するお店の情報表示画面へと遷移する．自分の水族館に追加ボタンをタップすると，自分の水族館に魚が追加される．

（文責: 田村修吾）

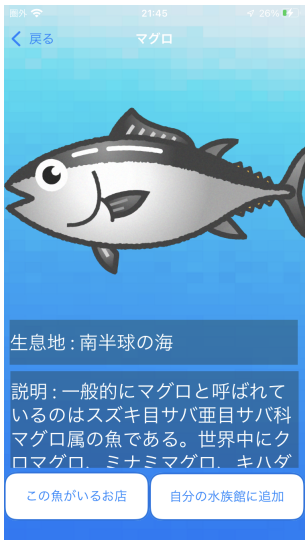

図 5.3　魚の情報表示画面

お店の情報表示画面

お店の名前，写真，主な取扱品，詳細な説明を表示する．これらの情報はすべて函館朝市のホームページから参照した [11]．参照するにあたって，函館朝市の方とコンタクトを取り，函館朝市のホームページの情報を利用する許諾をいただいた．

（文責: 田村修吾）

自分のおさかな画面

魚の情報表示画面で追加した魚をリスト表示で見ることができる．表示には UIKit の UITableView を利用した．今回は UITableView のセルを追加する機能，セルを表示する機能，セルを選択する機能を利用した．図 5.5 に表示されているように，画面の右上の削除ボタンをタップすると，確認画面の後，追加した魚をすべて消去できる．魚のリスト表示された部分をタップすると，追加したお魚画面へと遷移する．

図 5.4　お店の情報表示画面

（文責: 田村修吾）

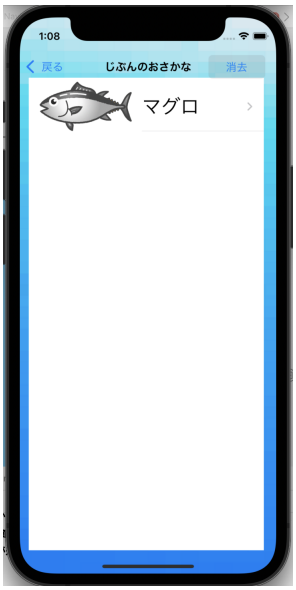

図 5.5　自分のお魚画面

追加したお魚画面

魚の名前，魚のイラスト，魚の生息地，詳細な説明に加えて，魚を追加した日時を表示する．魚の情報表示画面と同様に，UiKiT の UIScrollVeiw を利用した．UIScrollVeiw を利用することで，詳細な説明のように長い文章を限られた範囲内に表示させることができる．

（文責: 田村修吾）

図 5.6　追加したお魚画面

ガイド画面

「魚を探す機能」,「魚やお店の情報表示機能」,「自分の水族館機能」の 3 つの機能を説明する. 画面の表示には UIKit の UITableView を利用した.

（文責: 田村修吾）

図 5.7　ガイド画面

5.2.3　サーバーサイド・アプリケーション

本サービスでは，お魚やお店の情報変更ができるようにするために，サーバーを開発する必要がある．そのためサーバーサイドでは PHP を利用してを開発をした.

開発環境

- AWS EC 2
- PHP
- MySQL

データベース

本サービスのデータベースにはお魚の情報を表示する fish，お店の情報を表示する store の 2 つのテーブルが存在する．これらは MySQL を用いてデータの管理をして，PHP で JSON 出力することにより，データの送信ができるように開発した．また検知したビーコンによって表示する情報が異なるため，iPhone から，HTTP リクエストにある POST メソッドで送受信できるように開発した．これによりお店の情報が変更になったとしても MySQL にアクセスする事で，すばやくデータの更新ができる．それにより，新しいお魚を増やしたりする事が可能になる．

表 5.1　store テーブル

カラム名	型	概要
name	string	お店の名前
recommendation	string	おすすめの商品
uuid	int	ビーコンとの紐づける番号
introduction	string	PR 文

表 5.2　fish テーブル

カラム名	型	概要
fish	string	お魚の名前
habitat	string	生息地
Introduction	String	説明文
uuid	int	ビーコンとの紐づける番号

API

- GET/fish
  データベースに登録されているすべてのお魚の情報を表示する．なおお店の情報は iPhone から登録することはないため，サーバーサイド側で初期値としてすでに登録されている．
- GET/store
  データベースに登録されているすべてのお店の情報を表示する．なおお店の情報は iPhone から登録することはないため，サーバーサイド側で初期値としてすでに登録されている．
- POST/fish
  データベースに ID を送信する．送信した ID を元にしてデータを返す．
- POST/store
  データベースに ID を送信する．送信した ID を元にしてデータを返す．

（文責: 山口将馬）

5.2.4 デザイン

アプリケーションロゴ

アプリケーションロゴとは，スマートフォンのホーム画面に設置されるプリケーションのアイコンである．アプリケーションロゴは Canva を用いて作成した．水族館をなるべくイメージしやすく，かつ小さくなっても見やすいように心がけてメンバー間で相談しながら作成した．

（文責: 石田海祐）

アプリケーションデザイン

アプリケーションデザインは，まずは Figma でデザイン案を考え，それを Stortboard 上で再現した．全ての画面は海の中をイメージして制作した．ボタンは丸くして泡を連想させ，ほかに魚，岩，海藻などを描き海の中を連想させやすいようにした．

（文責: 石田海祐）

第 6 章　各メンバーのふりかえり

6.1　各メンバーの役割

- 山口将馬
  - プロダクトオーナー
  - サーバーやデータベース
- 中山寿似也
  - スクラムマスター
  - アプリケーション開発
  - UI の実装
- 石田海祐
  - アプリケーション開発
  - ロゴデザイン
  - UI デザイン
- 田村修吾
  - アプリケーション開発
  - UI デザイン

6.2　山口将馬のふりかえり

プロジェクト活動を振り返ってこの 1 年間で非常にさまざまな経験ができたと感じている．初めてサービスを開発する一連の流れというものを体験ができた．前期では，一番はじめにロゴ決めを行った．自分自身で何かのロゴのデザインを考えるという経験は初めてであった．簡単な 1 つのロゴではあるが意外と頭の中に浮かばないということを実感した．自分が制作したロゴは「函館の街といえば」というのを意識して海産物を取り入れたロゴを作成した．ほかの人も函館のシンボルである五稜郭公園の星形，五稜郭タワー，函館の海産物で有名であるイカなど，函館の象徴ともなるようなものをロゴとして取り入れている印象であった．最終的に決まったロゴは五稜郭タワーとカニを取り入れる形になったが，ここで重要なことに気付いた．実は函館ではカニが取ることはできないのである．この事実は前期の中間発表において指摘された．函館でカニが取れないというのは意外ではないだろうか．たしかに，よく考えればカニといえば長万部が有名であるというのは後になってから気付いた．自論ではあるが「函館を含めて道南の街を」と広く考えれば大丈夫であると考える．また，函館の街にある課題や問題を発見するためにフィールドワークを行った．私は五稜郭周辺グループであったので，五稜郭公園とシエスタハコダテに行った．五稜郭公園では COVID-19 の影響でいくつか閉鎖されている施設があった．たとえば五稜郭タワーはすべての階で立ち入る事ができなかった．そのほかの公園内の施設では箱館奉行所があった．ここでは入場料 250 円を払うことで入場することができた．館内の一部の閲覧物では使用制限されているものがあった．箱館奉行所を含めた五稜郭公園では，函館の歴史について深く学ぶ事ができた．また，公

園内は自然が豊かであり，ご年配の方などが散歩している様子が目に入った．シエスタハコダテでは主に無印良品，G スクエアに行った．時間帯のせいもあってか高校生が非常に多いと感じた．全体的に若い年代で賑わっているという印象であった．G スクエアではたくさんの掲示物の閲覧ができた．普段であれば壁に貼っている掲示物などを閲覧することは少ないが，フィールドワークの際はしっかりと掲示物に目を通すことで．函館の街で行われている行事などについてよく知る事ができた．しっかりと目的を持ってフィールドワークを行うことは初めてであったが，感想として，新しい発見をするということは想像以上に難しいと感じた．シエスタハコダテなどに行くと見慣れているせいか新しい発見をするということがなかなかできなかった．普段とは違う視点で物事に目を通す必要があることを痛感した．また少しでも意識をしていることで，何気ない発見ができた．五稜郭はメンバーが住んでいる地域が近いというせいもあってか，どこへ行きたいか投票をした際には人気がなかった．私は行ったことにある場所にフィールドワークに行くからこそ，あらためて新しい発見というものができると考える．行ったことない場所にフィールドワークに行ってもすべてが新しい発見であるため，普段では気付かないことというのは分からないと感じたためである．フィールドワークの後には miro や模造紙を使ってまとめを行い，そのことについて会話する事でその場では発見できかったようなことに気付くことができた．フィールドワークのまとめの目的はサービス開発のアイデアを出すことである．前期で一番苦戦した事はアイデア決めである．フィールドワークではいくつもの課題や発見などがあったが，そこからサービスのアイデアに持ってくということに苦戦した．まず，函館の街の課題を考えることから始まった．街の課題というものはさまざま案が出たが，その案とビーコンを組み合わせたアイデアを考えることが一番難しかったと感じる．先生も仰っていたが，ビーコンはアイデア勝負であるというものを実感した．開発で一番難しいのはアイデア出しであるということをここで初めて学ぶことができた．後期に入りサービスの最終決定するところで，ユーザーに提供できる価値，体験できることなど，どのような意図でそのサービスを開発するのかを決めることに時間をかけた．この時に決めたものは開発していく上で，方向性がずれないためにも非常に重要であったと，開発を終えてから感じた．サービスを開発する時には開発する前の土台固めが非常に重要であるとこのプロジェクト学習を通して体験できた．実際に開発に入ってからは，初めて共同開発を行ったので，分からないことがたくさんあった．私たちは部分ごとに分担してソースコードを書いていたが，何回かコンフリクトを起こしてしまった．コンフリクトとは同じファイルを編集してしまうことにより，ファイルの衝突が起きてしまうことである．一人で開発する時には気にしないことであるが，共同開発ではほかの人がどこのソースコードを編集しているかなどを考えながら開発をする必要があると感じた．また，私は前期の中間発表，後期の成果発表会で動画大臣として動画を制作した．私は動画制作をするのが好きで発表動画制作に携わりたいと強く感じたためである．動画制作では字幕を付けることが一番たいへんであると感じた．レビューでは感情的になってしまう面があった．これは私が今後改善しなければいけない部分である．今回函館朝市を盛り上げたいという想いでサービスを開発した．実際にアプリケーションをリリースして，使ってもらいたいと思ったがいくつかの課題があって難しいと感じた．私たちのサービスが実際に使われることになれば函館朝市を盛り上げることができると考える．私は，このように街を盛り上げるようなサービスの開発をまたしてみたいと強く感じている．また，私はプロダクトオーナーとしてチームを引っ張っていく存在であったので，もっと積極的に活動をすればよかったと反省している．今回ビーコンプロジェクトのグループは 3 チームあった．その中では活動時間がほかのチームに比べて少なかったと感じている．今回間に合わず未実装になってしまった部分がいくつかあったので，また機会があればプロダクトオーナーという立場で

チームを引っ張っていく存在になりたいと思った.

（文責: 山口将馬）

6.3 中山寿似也のふりかえり

前期のから夏休み終わりまでの活動は活動の進行方法の決定，フィールドワーク，アイデア出し，アイデアの絞り込みなどを行った．私たちは円滑に活動するにあたってプロジェクトリーダーを決めずに大臣制度という管理責任を持つ人を決めて活動することになり，ファシリテーターも全員が経験した．それにより話し合いを円滑に進めていくことの難しさや意見を引き出すことの難しさを学ぶことができた．フィールドワークは，事前にレクチャー動画での目的を持ってフィールドワークを行った．私は函館の駅担当で，函館駅や木古内駅の周辺，いさりび鉄道を主にフィールドワークを行った．コロナ禍ということもあり，駅や電車内でもコロナ対策が施されていて，利用している人も少なく感じた．電車内ではアナウンスの音が小さいことや，電車の利用方法がわかりづらいことなどを気付くことができた．これまでフィールドワークをしたことがなかったため，普段とは違う視点で周りの物事を見てみると今までにない気付きにつながることを学ぶことができた．またフィールドワークのまとめではほかの班とフィールドワークを行って気付いた課題や問題点を共有し，アイデアの種となるような発見をまとめた．その際，どのようにして収束していくのかを考えておかないとアイデアにつなげていくことは難しいということを学んだ．アイデア出しは，フィールドワークから発見した課題や問題点からアイデア出しを行った．アイデアの基準となる「ビーコンらしさ」「函館らしさ」「新規性」を満たすアイデアを考えることは非常に難しかったが，OST を活用してアイデアを選定していくことで円滑に進めることができた．アイデアの出しやその選定方法にもさまざまな方法があることを学ぶことができた．そのアイデアにはどのような新しい体験があるのか，ユーザーにとっての価値はなんなのかを考える必要があることを学んだ．中間発表前のアイデアの絞り込みでは，アイデア出しに時間がかかってしまい，ビーコンを使う意味や使用方法などの具体的な内容まで絞り込むことができなかった．アイデア出しや絞り込みに時間がかかってしまった理由として，自分の意見を出しづらい空気が全員にあったと感じる．そこで自分のアイデアを気軽に言えるような空気づくりが円滑な話し合いにつながることを学んだ．中間発表後にこれらのアイデアを 3 つに絞ることができた．その際に全員の意見のまとめ方について事前に判断基準を決めておくと絞り込みやすいということを学んだ.

夏休み期間には, 選定したアイデアの具体化や開発までの準備をした．アイデアの具体化ではアイデアごとのチームに分かれて機能や価値などについて話し合い，全体でレビューをした．アイデアを具体化していく際にメンバーとの食い違いが生じ，考え直すことが多くあったので細かいことでも小まめに確認してメンバーとの共通認識を持つことが大切だということを学んだ．開発までの準備では，チームのメンバー全員が開発未経験だったため，自分たちの作りたいものにどのような技術が必要なのかを選定することが難しく，開発期間に入ってから必要な技術が出てきてしまうことがあった．その際に作りたいものを細分化して必要な技術を細かく選定していくことが必要なことを学んだ.

後期では，主にアイデアの見直しや開発などを行った．アイデアの見直しでは前期や夏休み期間中に具体化したアイデアのレビューで得たフィードバックを元にアイデアを改善した．アイデアは，当初函館朝市内の魚の情報と，お店の情報を表示するアプリケーションの予定だった．レ

ビューから，ユーザーがアプリケーションを使いたい価値は何なのかを問われたときに，お店側からのメリットしか考えていないことに気付いた．そこで，函館にはない水族館をアプリケーション内で体験することによりユーザーに楽しんでもらえるというアイデアを生み出せた．レビューをすることでチームメンバーだけでは気が付かなかった改善点を見つけることができることを学んだ．開発では，スクラムマスターという役割を担った．スクラムマスターとはスクラムについてチームメンバーを先導していかなければいけない．しかし，開発経験もなくスクラム開発についての知識もまったくなかったため，スクラムマスターとしての役割をこなすことができるか不安があった．それを解消するためスクラムガイドやアジャイルワークショップなどを活用し，スクラム開発について学習することでスクラムマスターとしての役割をこなすことができた．デイリースクラムやスプリントレビューでは，最初は何のためにするのかを意識せずただただ形式的にこなしていただけだった．それを改善するため，プロジェクト内のスクラムマスターでチームの状況について話し合ったときに，ほかのチームからデイリースクラムやスプリントレビューなどを何のためにするのか考えた方が良いというアドバイスをもらい，より良いスクラム開発につなげることができた．スクラム開発を通して，チーム開発でのコミュニケーションの大切さを学ぶことができた．プログラミングでは，ビーコン検知機能を主に担当した．ビーコン検知機能はサービスの核となる機能であるため，絶対に完成させなければいけないプレッシャーがあった．ビーコンとはどのようにデータを処理して，アプリケーション内で扱うことができるようにするのか，ビーコンを検知した際の処理はどこに書くのかわからないことだらけだった．また開発経験がないため不安も大きかったが，書籍を読むことや先輩に聞くことなど自分から課題や問題を解決することの大切さを学んだ．開発を通しての反省点として，進捗状況に透明性がなくスプリントゴールを達成できないことがあった．できることできないことを自分で判断するだけでなくメンバーにも確認する必要があることを学んだ．そのほかには，アイデアの価値を詰めることができていなかったため，スプリントの中で変更しなければいけない点が出てきてしまったことがある．変更に対応するため，メンバーと会話しながらで作業しなければいけない時間ができてしまい，迷惑をかけてしまうことがあった．成果発表では動画を作成した．本サービスのユーザーストーリーをわかりやすく伝えるために函館朝市で撮影をした．実際にアプリケーションを使用しているように見せることでわかりやすい動画を作成できた．動画のナレーションでは聞き取りやすい声の大きさと速度で話すことが，見やすい動画を作る上で大切であることを学んだ．質疑応答では，自分たちの開発したサービスの紹介をした．サービスをわかりやすく簡潔に伝えるために，アプリケーションの画像を使い機能を説明した．その際に，自分たちの気がつかなかったサービスの良い点を指摘された．そこから客観的な視点でサービスを見つめ直すことで，新しい発見ができることを学んだ．

プロジェクト活動を振り返ると，さまざまな新しい経験ができた．函館の問題をビーコンを活用して解決するという活動は，将来に役立つ貴重な経験である．プロジェクト学習が始まった当初は，メンバーに知り合いが少なく，開発経験もないため不安だった．メンバーどうしが遠慮して，話し合いが進まないこともあったが，活動をかせねていくうちに自分の意見を言えるようになった．今思えばメンバーに助けられてばかりだったが，自分自身プロジェクト活動をする以前より成長できた．プロジェクト学習とメンバーに感謝し，問題解決能力やコミュニケーション能力など活動で身につけたことを今後の活動に活かしていきたい．

（文責: 中山寿似也）

6.4 石田海祐のふりかえり

今年度のプロジェクト学習は，COVID-19 の感染拡大がいまだ収束せず，オンラインでの活動が多くなってしまった．そんな中でも，対面での活動，フィールドワークが実施できたことは貴重な体験だったと思う．前期の活動は，主に全員で話し合って会議を進めることが多かったため対面の方が進行しやすかったと感じている．それに対して，後期の活動は開発が主な作業であったため各個人の家の環境で開発に取り組めるため，オンラインの方が進行しやすかった．前期の活動は，ファシリテーターをすべてのメンバーが経験した．私がファシリテーターを経験して思ったことは，その日のアジェンダを決めていたことなど，本来のファシリテーター以外のこともやっていた気がした．ファシリテーターとして会議を進行するのは想像よりも難しく，プロジェクトメンバーのレスポンスが必要不可欠であった．私はファシリテーターを経験する前は他のメンバーがファシリテーターをやっていても，十分なレスポンスを返してあげられていなかった．そのため，自分がファシリテーターを経験して感じたことや思ったことは，その後の活動では改善して取り組むことができた．前期の活動はアイデア出しがメインだった．ブレインストーミングで候補をいろいろ出してみることや，KJ 法など，これまでに私がしたことなかったことをたくさん経験できた．その中でも私は，フィールドワークが一番印象に残っている．フィールドワーク当日のこともちろん印象に残っているがどこへ行くか決める企画段階，日程を決める段階，プレフィールドワークの段階，フィールドワークが終わったあとのふりかえりまで，フィールドワークの一連の流れが印象に残っている．私はフィールドワークで道南いさりび鉄道を利用し木古内駅，また木古内駅の周辺を訪れた．函館駅から出発したいさりび鉄道車両内の様子は，平日であったためか，高校生が多い印象だった．西に進むに連れ高校生は減っていき残ったのはご年配の方々だった．木古内駅周辺は遊べる場所や学習をする場所が少ない印象だった．フィールドワークを通して，いつもと何も変わらない日常であっても，何か疑問に思うことや考えることが増えた．これによって前まではつまらないと思っていた瞬間も少しは楽しい時間に変えることができるようになった．フィールドワークが終わったあと，ふりかえりを行った．ふりかえりで気付いたことや思ったことなどをもとにサービス案を考えた．サービス案を考える中で OST という手法が印象に残っている．テーマを決めて雑談している中で何かアイデアを見出すという手法である．私自身，頭に思いついたことを言語化するのがあまり得意ではなく，サービス案を出すとなるとなかなか思いつかないなと思っていたが，OST を実行することによってアイデアが自然と頭に浮かぶようになった．夏季休業にはいる前に開発するサービスが 3 つに絞られた．3 つとも開発に携わってみたかったが，函館がより盛り上がると感じたのはあさいち水族館であると思い，あさいち水族館の開発シームに入った．夏季休業では GitHub の勉強会を行った．プロジェクトメンバーに開発経験がある人がいたためメンバーに教えてもらうことが多かった．GitHub は夏季休業だけでは使いこなせるようにはならず夏季休業が終わって本格的に開発が始まっても GitHub の使い方で詰まることが多かった．GitHub に関して，もっと自分で勉強しておくべきだった．後期の活動は，前期とは異なりオンラインでの活動が基本であった．更に活動内容もアイデアを絞り終わり開発するサービスが決まって開発が始まった．それに伴い，全体で活動することが減りほとんどはグループごとにブレイクアウトルームに分かれて活動していた．後期の活動で印象に残っていることは，やはり自分自身の技術面での成長である．3 年次にあがったばかりのころは開発経験があるどころか，プログラミングを書く機会も少なかった．しかしながら，グループメンバーや先輩のサポートがあり，自分が少しずつプログラミ

ングのコードをかけるようになっていったのを感じた．たとえば，開発し始めた当初は，アニメーションに力を入れる予定だった．しかしいざアニメーションの導入を試みるとなかなか思うようにならずに苦戦してしまっていた．そのときにチームのメンバーが一緒に考えてくれて問題を解決することができ無事にアニメーションを導入することに成功した．このように開発を通してメンバーとの協調性を培うことができた．私はサービスのデザインをメインで担当していて，ボタンの細かな配置や形，UI の設計などを行った．これまでの人生でデザイン系は触れてこなくて，初めての挑戦であったが，おおむね満足行くデザインになったと考えている．後期の活動ではグループでの活動がメインになり，責任の所在が前期の活動より明確になった．それに伴いプロダクトオーナーへの負担が大きくなってしまったのが反省点として挙げられる．プロダクトオーナーにもっと献身的になり話し合いに参加したり，提案したりできたら良かったと思う．自分の手が空いている時間をもっと有効活用できていたらより良い活動になって，より良いサービスが作れたのではないかと感じた．前期の活動では主に積極的に思考して自ら動いたりする行動面，後期の活動からは主にプログラミングの技術面と，自分自身の成長が肌で感じられる一年になったと思う．プロジェクト学習でできなかった反省点を改善し，一年間で得た人間性，プロジェクトマネジメントスキル，プロジグラミング力を今後の卒業研究，自主的に行う活動，就職してからの会社への貢献の活動に活かしていきたい．

（文責: 石田海祐）

6.5　田村修吾のふりかえり

今回のプロジェクト学習を通して，これまでの生活では経験できなかったことを多く経験できたと感じる．これまでチームでの開発やサービスの考案，アプリケーションの開発という経験もなかった．そのため，アイデア出しからアイデアの絞り込み，開発やレビューなどの経験がとても新鮮で，難しいと考えることも多かったが，楽しいと感じた．前期では，ロゴやサービスのアイデアを出し合い，絞り込み，決定していく活動が主な活動だった．フィールドワークを行い，函館のまちの課題を発見，プロジェクト内で共有し，サービスアイデアの種となるものを整理した．フィールドワーク後すぐにふりかえりを行うことで，その時には自分では気が付かなかったことをほかのメンバーと共有することで，発見できた．アイデアの種をもとに，発見した課題を解決するサービスのアイデア出しを行った．出されたアイデアの絞り込みを主に「函館らしさ」,「新規性」,「おもしろさ」の 3 つの価値基準を用いて行った．これらを OST などを利用しながら繰り返し行った．このとき，今は収束と発散のどちらを行うのかを決めながら，アイデアの収束と発散を繰り返し行うことで，より精査されたアイデアをプロジェクトメンバー全員が納得する形で決定できた．また，ファシリテーターを全員が体験し，話し合いを円滑に進め，複数人の意見を引き出し，まとめることの難しさと苦労を学ぶことができた．特に Zoom を用いたリモート活動では，1 つのチャンネルにプロジェクトメンバー全員が集まって話し合うのではなく，ブレイクアウトルーム機能を利用し，3 人程度の少人数で話し合いをし，結果を全体に共有するという形式を取ることで，円滑な話し合いができることを学んだ．このように少人数では話し合いがスムーズに進むが，全体の場では話し合いがしづらいといった問題の原因として，プロジェクトメンバー各々が意見を述べづらい環境があると考えた．このことから，メンバー一人一人が自分の意見を述べやすい環境を作ることが話し合いでは重要であると学んだ．後期では，各サービスチームでの開発が主な活動だった．開

発を始める前に，提供するサービスの見直しや修正をし，ユーザーが得られる価値やサービスの内容を決定した．ここで決定したものをサービスの核とすることで，開発が進めやすくなり，今後重要なポイントであったと感じた．開発が始まってからは，私は主に UI のデザイン，設計を担当した．これまでにデザイン設計の経験がなかったため，試行錯誤を繰り返しながらの開発だったが，扱いやすいデザインを制作できたと考える．反省点として，チームメンバーの進捗や全体を通しての状況を把握できていなかったことが挙げられる．原因は，チームメンバーとのコミュニケーションが不足していたことであると考える．チームメンバーとのコミュニケーションが不足していたことで，共有するべき情報が共有できず，何かトラブルが起こった際に，チームメンバーへの確認ができていなかったと考えた．それにより，開発作業が滞ってしまい，スプリントゴールを達成できない事があった．このことから，チーム開発でのコミュニケーションとメンバー間との情報共有の重要性を学ぶことができた．1 年間の活動をふりかえり，メンバーに迷惑を掛けてしまったとともに，経験したことのなかった新しい学びを多く得ることができ，大きな成長の機会を得ることができたと考える．本プロジェクトの先生や TA，OB，OG の方々，特にプロジェクトメンバー，開発メンバーには感謝したい．1 年間で得た学びと成長を今後の活動や生活に活かしていきたい．

（文責: 田村修吾）

第 7 章　まとめと展望

7.1　前期のふりかえり

前期では主に 3 つのことを行った．1 つ目はロゴ決めである．ロゴ決めでは，全員がロゴ案を持ち寄ってそれぞれがロゴに込めた意味などを発表して，レビューを繰り返し最終的には多数決で決定に至った．何度もレビューを繰り返すことで，より良いものになっていった．2 つ目はフィールドワークである．フィールドワークでは函館駅周辺，函館近郊の駅，五稜郭周辺，湯川周辺 4 つのグループに分けて行った．フィールドワークを実施する前には南部先生によるフィールドワークのレクチャーを受けた．フィールドワークの実施中には，気になった点などはグループごとに Slack のチャンネルを作成し，随時文章や写真で Slack に投稿する形で記録した．COVID-19 の影響か，観光地や駅周辺などは人が少ない印象であった．3 つ目はアイデアの決定である．フィールドワークの実施後には miro や模造紙などを用いて，19 個のテーマごとにまとめた．その後，ブレインストーミングを行いアイデアを出していった．ブレインストーミングとはそのテーマについてメンバーで雑談などを踏まえながら議論する事である．議論というよりは会話するという方が表現的には近い．気軽にテーマについて会話することで，柔軟にアイデアを出すというものである．最終的に 7 個のアイデアが出た．7 個のアイデアを最終的には「函館らしさ」,「ビーコンらしさ」,「新規性」,「おもしろさ」の 4 つの価値基準で順位づけを行い，上位 3 つのアイデアを最終的なアイデアとして決定した．前期の中間発表の時には，アイデアを絞るところまでは進んでいなかったので 7 個のアイデアをすべて発表した．制作物ではポスターと動画を作成した．ポスターと動画ともにレビュー，修正を繰り返し見やすいものになるように心がけた．7 月ごろまでは，進行役を決めないで週替わりで進行役を行っていた．これにより全員が進行役の気持ちを理解ができた．また，誰が進行役に適しているかを決めることができた．7 月以降はメンバーの中で進行役に適している人をファシリテーターとして定め，円滑にプロジェクト学習を進めることができた．

（文責: 山口将馬）

7.2　後期のふりかえり

後期に入る前の夏休みに Git，GitHub の勉強会を行った．初めて使う人が多かったので，導入方法や，使い方について学んだ．後期に入ってからは最初に，グループ独自でフィールドワークを実施した．前期のフィールドワークでは函館駅周辺のグループが函館朝市に訪れていたが，ほかの場所にも行っていたため，滞在時間は少なくあまり時間を取ることができなかった．また，グループが異なり函館朝市に行っていないメンバーもいた．そのため，フィールドワークを実施した．フィールドワークの実施中は LINE を利用して随時記録をしていった．その後まとめを行った．その結果，函館朝市には地元の人が少なく観光客らしき人が多数である，そして店員さんも観光客を前提としていることが多いという印象であった．またお店がたくさんあり，どこに何があるかわかりにくいという印象もあった．その後決めたアイデアからサービスの最終決定をした．サービスが

ユーザーに提供する価値，ユーザーが体験できることなど開発に入る前に必要なことを決めた．私たちのグループではこれらを決めることに苦戦した．その時はフィールドワークを実施したときの記録などをもう一度見返して，ユーザーの立場に立って何を提供できればユーザーは楽しむことができるかなどを考えた．サービスの機能を決める時にはこの決めたことを基準にして，必要な機能などを決めていった．その後プロダクトオーナー，スクラムマスターなどの役割を立候補制により決めた．私たちのグループでは，サービスの最終決定をするにあたり，シーケンス図を決めることによりサービスを具体化していった．これは何をするかわからなくなってしまっていた私たちのグループへの TA からのアドバイスであり，図にして目で見えるようにサービスを具体化していった．これらの経緯などを踏まえ私たちはサービスの最終決定をした．定期的に地元の人が函館朝市に来てもらえるようにと私たちは自分の水族館機能というものを考案した．開発に入ってからは，1 週間もしくは 2 週間を 1 スプリントとして，1 スプリントごとにレビューをした．スプリントごとに目標を決めて開発に取り掛かっていたが結果的にはなかなか予定通りに進まず，スプリントがずれていってしまった．原因としては，今まで触れた事ない分野をやる人が多数でわからない点がいくつも存在していた事が挙げられる．そのためいくつか実装できなかった部分が存在してしまった．レビューでは，グループ内では気付かなかった点などを指摘してもらったり，意見などをもらうことによりサービスの質を向上させることができた．またスプリント後にはレトロスペクティブを行いその前の週のスプリントふり返っていた．そのほかには，デイリースクラムを行いその日に何をしたかなどのふりかえりを行った．これにより，誰がどんな活動をしたのかを毎日知ることができ，進行具合などの確認ができた．

（文責: 山口将馬）

7.3 展望

私たちが実装したサービスの現段階ではいくつかの課題や未実装部分が存在する．まず大きな課題として水族館要素が少ないという点が挙げられる．私たちは函館にはない水族館をアプリケーションで擬似体験してほしいという想いであさいち水族館というサービス名にした．そして当初の目標ではビーコンを検知すると水槽の中に検知したお魚が飛び込んでくるというアイデアを取り入れようとしていた．しかし実際のサービスはビーコンを検知するとボタンが出現するだけというものになってしまった．また，水槽の背景を本物の海にように波を演出したかったが実際には海の画像だけになってしまった．私たちのサービスでは，最初のメイン画面で魚を泳がせるために，lottie-ios というライブラリを利用したが，ビーコンを検知した際にお魚が飛び込んでくるという機能もこのライブラリを利用して実現させようとしていた．Adobe の Iiiustrator で絵を描いて AfterEffect で編集をすることでアニメーションを作成するところまではできた．しかしそのアニメーションをボタンにするところでうまくいかなかったというのと，時間がなくて完成させる事ができなかったでのある．時間がなくなった要因としては，最初の段階でスプリントが伸びてしまったりしたことである．グループ内で議論をしている時にアニメーションに力を入れるのではなく，自分の水族館機能に力を入れた方が良いのではないかという意見になったためである．また，今回未実装であった部分として，期間限定のお魚機能，お知らせ機能というものが存在する．自分の水族館機能では，自分のスマートフォンのアプリケーションの中にお魚をコレクションすることにより定期的に足を運んでもらおうという考えであった．しかし，スプリントごとに行うレビュー

の中で，すべての魚をコレクションしてしまったら定期的に足を運んでもらうことができなくなってしまうのではないかという意見が出た．そこで私たちは期間限定のお魚を出現させることにより，コンプリートをしてしまう事がなく定期的に足を運んでもらえるのではないかと考えた．また，期間限定のお魚を出現させることにより，それらのお魚を出現したことを知らせるお知らせ機能というものを実施しようとしていた．お知らせ機能で定期的にスマートフォンに通知を出すことにより，アプリケーションを忘れ去られることを防ぐ目的や，朝市に足を運ぶきっかけを作ることができると考える．もともとこの 2 つの未実装の部分は，途中で出てきたアイデアだったため，自分たちのスプリントがすべて終わって余裕があったら実装するというものであったので，今回スプリントが伸びてしまったりした影響から実装するには至らなかった．また，ほかの案として自分の水族館に餌やりをする機能というものを考えてた．この機能は函館朝市にいない時でもアプリケーションを楽しんでもらうという狙いがある．函館朝市でしか使えないサービスとなってしまうと短期間でアプリケーションが消されてしまうかもしれないという可能性があるためである．また，自分のお魚を育てることにより，お魚に愛着が湧いて，もっとお魚を増やしたいと思うユーザーが増えるかもしれないと考えた．お魚を成長させる育成ゲームなどの機能の拡張にもつなげることができる．自分の水族館機能をもっと充実させることにより，ほかのバーチャル水族館などのサービスと差別化を図りユーザーを獲得できると考える．これらの課題，未実装部分を解決ができればより多くの人に私たちのサービスを楽しんでもらえる事ができるのではないかと考える．

（文責: 山口将馬）

参考文献

[8] **北海道 経済部 観光局 観光振興課 / 令和 *2* 年度北海道観光入込客数調査報告書**. https://www.pref.hokkaido.lg.jp/kz/kkd/irikomi.html.

[9] **国土交通省観光庁｜統計情報・白書｜旅行・観光消費動向調査**. https://www.mlit.go.jp/kankocho/siryou/toukei/shouhidoukou.html.

[10] **函館市｜「観光動向調査」・「観光アンケート」調査結果｜**. https://www.city.hakodate.hokkaido.jp/docs/2014060600023/.

[11] **函館朝市オフィシャルサイト**. http://www.hakodate-asaichi.com/. (Accessed on 1/18/2022).

第 III 部

Favor について

第 1 章　背景

令和 2 年より国内で流行している COVID-19 の影響で函館の観光業界は大きな打撃を受けてしまった [12]．北海道新型コロナウィルス対策本部では各事業者や市民に感染対策の呼びかけを行っている [13]．感染対策としては特に飲食店での黙食や観光施設，公共交通機関での会話の制限がされるようになった．函館市は観光資源が多く，おいしい海産物を食べるために飲食店を利用したり，歴史的な建造物を見るために訪れる観光客が多い．しかし，COVID-19 の対策による黙食や会話の制限は，観光客や飲食店を利用する人のコミュニケーションがしにくくなっている．会話が少なくなると雰囲気が重たくなり制限されている場所で純粋に楽しむことが難しくなってしまう．そのため，本グループはコミュニケーションの制限による悪化する雰囲気を良くすることを課題とした．

（文責: 山内彩土）

第 2 章　目的

本グループの目的は，会話以外での新しいコミュニケーションの形を実現することで背景に挙げた課題を解決することである．これにより，会話の制限されている場所でもコミュニケーションを楽しく取ることができ，周りの環境や雰囲気の改善ができる．

（文責: 山内彩土）

第 3 章　サービスの概要

3.1　サービスの概要

「Favor」は，会話が制限されている中でのコミュニケーションを楽しみ，雰囲気を改善するための音楽共有アプリケーションである．同じビーコン範囲内にいる利用者どうしで Spotify の音楽を共有することでお互いのお気に入りの曲を知ることができ，一種のコミュニケーションを取る事を可能にしている．友人どうしだけでなく範囲内にいる知らない人どうしでの音楽共有ができるため，音楽を通じた一期一会を感じることができる．既存の音楽配信サービスでは QR コードや URL を送りあうことで音楽共有を可能にしている．しかし，本サービスではビーコン範囲内であれば音楽共有を可能にしているため他人に QR コードや URL を教えるなどの煩わしさを軽減している．これにより周りにいる人どうしでの音楽共有をする事を簡単にしている．

（文責: 山内彩土）

3.2　アプリケーションの機能

本サービスの機能は，「音楽検索機能」「音楽再生機能」「お気に入り機能」の 3 つに分けられる．「音楽検索機能」では，Spotify の楽曲を検索し，同じビーコン範囲内での共有プレイリストに楽曲を追加できる機能である．これによりユーザーは好きな楽曲を共有できる．この機能の検索対象は楽曲名とアーティスト名で，アルバム名を検索できない．これは検索対象の優先度を考えたとき共有するのはアルバム単位ではなく楽曲単位なので優先度が低いと判断したためである．「音楽再生機能」は，共有プレイリストに追加されている楽曲を順に再生する機能である．この楽曲再生機能は周りのユーザーと同時再生されているわけではなく各ユーザーが好きなタイミングで再生できる．音楽再生の同期をすると後からビーコン範囲内に入った人が最初に共有された楽曲を聞くことができなくなる．そのため，共有プレイリスト内の音楽再生は非同期とした．「お気に入り機能」は，ユーザーが「音楽再生機能」で再生された楽曲を気に入った時に利用する機能である．共有された楽曲にお気に入り登録をすることで共有をしているほかのユーザーに通知を送ることができる．これにより周りの人にお気に入りを伝えることができ，音楽を通じたコミュニケーションを楽しむことができる．

（文責: 山内彩土）

第 4 章　開発

4.1　技術取得

チームメンバー全員が SwiftUI を使った iOS 開発は初めてだったため，チームで SwiftUI の勉強会を行った．勉強会には，Apple が掲載している公式の SwiftUI チュートリアル [14] を利用した．チュートリアルのすべてをやった訳ではなかったが，SwiftUI がどのようなものか理解できた．

（文責: 佐々木紀明）

4.1.1　利用したサービス・ツール

チーム開発において，アイデアや問題点などの共有，チーム内でのコミュニケーションは非常に重要である．また，開発やデザインの決定にさまざまなツールを利用した．以下に利用したサービス・ツールを示す.

（文責: 佐々木紀明）

Slack

Slack とは，人々をそれぞれが必要とする情報につなげる，ビジネス用のメッセージングアプリケーションである [15]．チームメンバー間や教員，TA，ほかサービスのメンバーでの連絡に Slack を利用した．Slack では，ファイルや URL などの共有を容易に行うことができる．また，チャンネル登録をしているメンバーは全員がメッセージのやりとりを見ることができ，チーム外の活動も確認ができた.

（文責: 佐々木紀明）

Discord

Discord とは，アメリカで誕生したボイスチャットサービスである [16]．チームメンバー間の相談や，デイリースクラムなどに Discode を利用した．メンバー間での通話を容易に行うことができ，画面の共有などの機能があり円滑に会話を進めることができた.

（文責: 佐々木紀明）

Zoom

Zoom とは，先進的なユニファイドビデオコミュニケーションプラットフォームである [17]．システム情報科学実習の時間にプロジェクトメンバー全員でコミュニケーションをとるために Zoom を利用した．Zoom ではカメラを利用することでお互いの表情を確認しながら会話ができる．ブレイクアウトルームを作成し，各チームごとに分かれての活動もできた．また，チャット機能やリア

クション機能などもあり，メンバー間でのコミュニケーションを取りやすかった．

（文責: 佐々木紀明）

esa

esa とは，情報を育てるという視点で作られた自律的なチームのための情報共有サービスである [18]．日々の活動を議事録として残すために esa を利用した．esa では，複数人で記事を同時に編集できるため，メンバーで協力しあって記事を書くことができた．esa に活動内容を残すことで活動内容や次回やるべきことなどをいつでも確認できた．

（文責: 佐々木紀明）

Miro

Miro とは，ビジュアルコラボレーションプラットフォームである [19]．チーム内の話し合いなどを円滑に進めるために，Miro を利用した．Miro では，さまざまなテンプレートや機能があり，アイデア出し・ユーザーストーリー作成・スプリントレトロスペクティブなどに利用できた．

（文責: 佐々木紀明）

Figma

Figma とは，アメリカ発の UI デザインツールである [20]．アプリケーション内のデザインやロゴ案の制作に，Figma を利用した Figma には動線設計，画面設計，デザイン，プロトタイプ制作に利用できる多くの機能が備わっている．そのため，デザイン制作を容易に行うことができた．

（文責: 佐々木紀明）

Adobe Illustrator

Adobe Illustrator とは，ベクターベースのグラフィックデザインソフトウェアである [21]．アプリケーションのアイコン作成に，Adobe Illustrator を利用した．主にベクター形式の図形や，イラスト，デザインなどを作成できる．デザイン作成などに便利な機能が多く備わっているため，容易にロゴを作成できた．

（文責: 佐々木紀明）

Git

Git とは分散型のバージョン管理システムである [22]．ソースコードのバージョンを管理するために Git を利用した．リポジトリの中にあるファイルの変更履歴と完全なコピーを保存できる．これにより，どのような変更が行われたかを確認でき，それをもとにしてファイルを変更前の状態に戻すこともできる．また，Git には branch という概念が存在し，複数のメンバーで同時に作業できる．このように Git には作業を便利にする多くの機能がある．

（文責: 佐々木紀明）

GitHub

GitHub とは開発プラットフォームである [23]．GitHub を使用することで複数メンバーでの開発作業を円滑に進めることができる．GitHub には多くの機能があるが主要な機能として Pull Request が挙げられる．これは，分岐した branch を分岐元へ取り込む際に使用される．この機能を使うことでいつ誰がどのファイルを変更したかを把握できる．また，Pull Request をする際に理由やレビュー観点を記入できるため，ほかのメンバーからのレビューを簡単に受けることができる．私たちのグループではほかにも GitHub Projects という機能を使いタスク管理なども行った．GitHub Projects とはカンバン方式のタスク管理機能で，「現在どのタスクが進行しているか」や「そのタスクがどのような状態であるか」などを把握できる．

（文責: 佐々木紀明）

4.2 開発手法

4.2.1 導入した開発手法

アジャイル

本アプリケーションの開発にはシステムやソフトウェアの開発手法の 1 つである，アジャイル開発を導入した．アジャイル開発とは，短い期間での実装とテストを繰り返すことで製品の質を高める方法である．これにより開発する優先順位をつけやすく，簡単に軌道修正ができる．また，前期にアジャイルワークショップへ参加していたこともあり，この開発手法を採用した．

（文責: 佐々木紀明）

スクラム

本アプリケーションの開発にはアジャイル開発手法の 1 つであるスクラムを採用した．チームメンバーが 3 人であったためプロダクトオーナーやスクラムマスターが開発を兼任した．チームメンバーは全員スクラム開発の未経験者であったため，最初のスプリントではスクラムに慣れるため 2 週間とした．その後はほかのサービスとレビュー時期を合わせるために 1 週間とした．4 週目のスプリントから GitHub Projects を利用してスプリントプランニングとタスクの管理した．デイリースクラムは，昼に行うことで余裕を持って集合できた．しかし，開発をする中で生活リズムが乱れていることに気付き，デイリースクラムの時間を朝に変更し生活リズムを整えた．また，スプリントレトロスペクティブでは KPT 法を採用して行った．

（文責: 佐々木紀明）

第 5 章　実装

5.1　システム構成

5.1.1　SpotifyAPI

本アプリケーションの開発では，Spotify が公開している API を用いた．Spotify は，API を介して楽曲再生や，プレイリストを編集できる．本開発では，Spotify の Web API を用いて実装した．サービスとして Spotify を採用した理由は，メジャーなサービスやソースの多さ，メンバーがサービスの使用経験があるという 3 点を考慮し，採用した．メジャーなサービスという点では，本アプリケーションはメジャーな楽曲を想定しているため考慮した．ソースの多さという点では，メンバーの開発経験があまりなかったため，開発の取り組みやすさという点で考慮した．サービスの使用経験がある点では，開発をしていく上で，サービスの仕様を理解するというコストがなくなるため，考慮した．

使用した API は, プレイリストの操作, 楽曲情報の取得, Spotify アカウントの参照を目的として利用した．リクエストは，各リソースに対してエンドポイント，HTTP メソッドを設定し，送った．送ったリクエストに対して，レスポンスとして JSON 形式で返ってきたデータを用いて出力した．リクエストの内容は以下の通りである．以下では，baseAPIURL を https://api.spotify.com/v1 とする．

プレイリストに関する API

- GET baseAPIURL/playlists/playlist.id
  操作するプレイリストのプレイリスト名，オーナー名を JSON 形式で返す.
- GET baseAPIURL/playlists/playlist.id/tracks
  ユーザーが，操作するプレイリストに入っている楽曲情報を JSON 形式で返す.
- DELETE baseAPIURL/playlists/playlist.id/tracks
  ユーザーが，操作するプレイリストに入っている楽曲を削除する.

（文責: 萩原啓道）

楽曲検索に関する API

- GET baseAPIURL/search?limit=30&type=artist,track&q=query
  ユーザーが，任意の楽曲を検索する際に使用する．Spotify にある楽曲名やアーティスト名を検索結果として，30 件の楽曲を返す.

（文責: 萩原啓道）

ユーザー情報に関する API

- GET GET baseAPIURL/me
  ユーザーの表示名，UserID，利用している Spotify の Plan を，JSON 形式で返す.

（文責: 萩原啓道）

5.1.2 デザイン

サービスデザイン

サービスデザインを決めるにあたって，初めに必要な機能を決定した．決定した機能としては音楽共有機能，音楽検索機能であった．最低限必要な機能を決定し，3 回目のスプリントレビュー時に曲を共有するだけで周りとのコミュニケーションができるのかという課題が見つかりお気に入り機能というユーザーのコミュニケーション手段になるような機能を必要な機能として追加した．ビーコンの検知機能についても優先して実装する機能として設定した．

UI などのデザインは主に Figma を用いて作成した．Spotify の API を用いたサービスになるため UI を Spotify に寄せて設計した．カラーデザインは Spotify のメインカラーを参考にしつつ，同じにならないように決定した．また，Spotify の背景色は黒だが Favor については白とした．これも差別化を図るためである．これによりユーザー側の操作がわかりにくくならないようにした．

（文責: 山内彩土）

サービスロゴ

ロゴを制作するにあたり，サービスの特徴や要素を考えた，音楽，共有，お気に入り登録などの要素が挙がりそれらを含めたロゴデザインを考えた．無料のロゴ制作ツールなどを利用する案も挙がったが自分たちで考えたほうが自由に制作できることから Adobe Illustrator で制作することにした．音楽アプリケーションであることを示すため，ロゴマーク全体が音符の形になっている．サービス内の機能でほかの人が共有した曲をお気に入り登録するというものがあり，その際にハートマークのボタンを押すため，ロゴの中にもハートマークを入れた．音符マークを構成していることに 3 つの丸を線でつなぐことによって人と人とのつながりを表している．背景の色はサービス内の背景と同じ色になっている．Favor の文字とロゴマークの位置関係やロゴマークの色などはメンバーと話し合いながら決定した．

（文責: 佐々木紀明）

5.1.3 モバイルアプリケーション (iOS)

開発環境

- Xcode
  本開発では，iOS アプリケーションの開発を実施した．iOS アプリケーションの開発では，Xcode を使用した．Xcode は，Apple 社が提供している統合開発環境である．使用したバージョンは，13.1 である．

図 5.1　ロゴ案 1

図 5.2　ロゴ案 2

図 5.3　ロゴ案 3

図 5.4　ロゴ決定案

- Swift
  Swift は，Apple 社が開発したプログラミング言語である．iOS や Mac，Apple TV，Apple Watch 向けのアプリケーションを開発するための言語である．使用したバージョンは，5.5.1 である．
- CocoaPods
  CocoaPods は，Swift および Objective-C 向けのライブラリ管理ツールである．本開発は，外部ライブラリを使用したため，CocoaPods を使用した．使用したバージョンは，1.11.2 である．

（文責: 萩原啓道）

使用したライブラリ

本アプリケーションでは，ライブラリを管理するツールとして，CocoaPods を用いた．本開発では以下の 3 個のライブラリ，フレームワークを導入した．導入したライブラリは，以下の通りである．

Appirater

本アプリケーションでは，ログインした Spotify をサインアウトする際に用いるライブラリである．サインアウトの際は，アラート表示を出力し，サインアウトする．サインアウト機能を実装する際，簡単に行えるため，このライブラリを使用した．使用したバージョンは，2.3.1 である．

SDWebImage

楽曲情報の取得，そしてユーザーデータの取得をする際に画像を非同期でダウンロードする．また，ダウンロードした画像のキャッシュを行うライブラリである．非同期処理など，このライブラリを使用することで，より簡単に画像を利用できるためこのライブラリを使用した．使用したバージョンは，5.12.1 である．

AVFoundation

プレイリスト内の楽曲を再生する際に使うライブラリである．本アプリケーションでは，Spotify の API から取得した楽曲情報を用いて，楽曲再生した．

（文責: 萩原啓道）

各機能の実装方法

- ログイン, ログアウト機能
  ログイン機能は，Spotify の OAuth2.0 認証を用いた．ログアウト機能は，Appirater を用いて実装した．API 通信に必要となる access_token，refresh_token，expirationDate の Value を空にすることで，ログアウト機能を実装した．

  （文責: 萩原啓道）

- 楽曲検索機能
  Spotify の API を用いて実装した．ユーザーが，打ち込んだ検索内容を，API を用いて Spotify 上で検索し，検索結果から出力をした．

  （文責: 萩原啓道）

- 楽曲再生機能
  AVFoundation，Spotify の API を用いて実装した．再生する楽曲の URL を Spotify の API を用いて取得し，AVFoundation で楽曲を再生した．

  （文責: 萩原啓道）

- Spotify アカウント情報閲覧機能
  SDWebImage，Spotify の API を用いて実装した．アカウント情報は，Spotify の API から取得し出力した．アカウントの画像は，Spotify の API から画像の URL を取得し，SDWebImage を用いて出力した．

  （文責: 萩原啓道）

- ビーコン識別機能
  後述するサーバーサイド・アプリケーションを用いて行った．

  （文責: 佐々木紀明）

- プレイリスト接続機能
  ビーコンとプレイリストを紐づけて, ビーコン接続と同時に対応するプレイリストへ接続できるようにした．

  （文責: 佐々木紀明）

5.1.4 サーバーサイド・アプリケーション

Favor チームから依頼があったため，FLAT チームの山本が担当した．モバイルアプリケーション側でビーコンを検知すると，API サーバーにビーコンの情報が送信される．送信されたビーコンの情報をもとに，ビーコンが設置してある場所の名前を返却するしくみになっている．実装はIV部 5.3 節を流用した．

（文責: 山本竜生）

5.2 実装した機能

5.2.1 Spotify アカウントへのログイン・ログアウト機能

ログイン機能は，各自の Spotify のアカウント ID とパスワードを入力することで，本アプリケーションの機能を利用できる．ユーザーが，各自の Spotify アカウントでログインすることで，Spotify の API を利用できる．ログアウト機能は，アカウント情報確認機能から行うことが可能である．アラート表示の後，実行後はログイン画面へと遷移する．

（文責: 萩原啓道）

5.2.2 音楽検索機能

検索タブにある入力欄へ，検索したい内容を入力する．入力した後は，Spotify 上にあるデータから検索し，表示する．検索対象は，楽曲名，アーティスト名のみである．検索対象の優先度を考えた際に，2 つの優先度が高いと考えた．そのため，本アプリケーションでは，アルバムとプレイリストを検索対象から外した．

（文責: 萩原啓道）

5.2.3 楽曲再生機能

楽曲再生は，再生タブにあるプレイリストから行うことができる．ユーザーは，プレイリストにある再生ボタンから楽曲を上から順番に再生する．Spotify の Web API の仕様上，現状はプレビュー再生のみとなっている．よって，現状の本アプリケーションでは，楽曲再生は 30 秒間の再生となっている．

（文責: 萩原啓道）

5.2.4 ビーコン検知機能

アプリケーションを起動した状態でサーバーへ登録されているビーコンの範囲内へ入ることでビーコンが自動的に識別される．複数のビーコンの範囲内にあった場合，スマートフォンに最も近

いビーコンが識別される．識別されたビーコンに対応したプレイリストが画面に表示され，曲が再生できるようになる．

（文責: 佐々木紀明）

第 6 章　各メンバーのふりかえり

6.1　各メンバーの役割

本チームでは，スクラム開発手法にのっとり，プロダクトオーナー 1 人，スクラムマスター 1 人を決定した．しかし，チームメンバーが 3 人しかいないためプロダくとオーナーとスクラムマスターも開発に参加した．3 人の役割は以下に示す．また，サーバーサイドアプリケーションの開発はほかチームのメンバーに有識者がいた為，作成を依頼した．

山内彩土

- プロダクトオーナー
- UI デザイン開発
- iOS アプリケーション開発

佐々木紀明

- スクラムマスター
- アプリケーションロゴデザイン作成
- iOS アプリケーション開発

萩原啓道

- UI の実装
- Web API との連携
- 開発環境整備
- iOS アプリケーション開発

（文責: 佐々木紀明）

6.2　山内彩土のふりかえり

プロジェクト活動を通して多くのことを学び，経験できた．前期では，活動ごとの話し合いの進め方やフィールドワーク，サービス考案のプロセスなどを学ぶことができた．特に活動ごとに行った話し合いでは，自分自身発言すること自体は少なくなかったが，発言の少ない人に意見を求めることはあまりできなかった．自分が発言することだけでなくほかの人の意見も聞けるような環境にすることで質の高い話し合いを行えると考える．

後期では，スクラム開発をした．初めてのスクラムでプロダクトオーナーを担当した．開発期間の開始当初は勝手が分からずスプリントゴールが適切なものでなかった．しかし，スプリントを重ねて適切なゴールを設定できるようになった．また，チームの人数が少なかったこともあり開発作業をすることもあった．本来のスクラム開発ではプロダクトオーナーは開発をすることはないが，問題に対して柔軟に対応できた．FLAT チームの資料を参考にしてプロダクトオーナーの役割や

スクラムでの活動を学ぶことが多かった．何も考えずに資料どおり進めることはしないよう自分で調べつつ活動した．しかし，FLAT チームの開発の流れをなぞるようになったのは否めない．そのため，もう少し自分のチームで開発の流れを工夫できたと感じる．スプリントレビューでは指摘されることがとても多く，サービスの根幹が揺らぐようなものもあった．たとえば，Spotify の規約に想定していた利用方法が禁止されていたことである．当初は飲食店などの BGM をジュークボックスのように店内のサービス利用者が共有できるようなサービスを考えていたが，スプリントレビューで規約に違反していることが判明し，現在のサービスの内容に落ち着いた．これは，規約の見逃しをしてしまったことが原因である．この一件から規約など，サービスに関わる重要な要項はしっかりと確認することを心がけるようになった．

開発後期には，Spotify の API を用いる機能やビーコン検知機能といった重いタスクが多くなってしまい，開発メンバーに負担をかけてしまうことが多くなってしまった．これはスプリントゴールの設定やスプリントプランニング時に適切に動けていなかったことが原因だと考える．チームの人数が少ないことを考慮して重いタスクを分割したり，できるだけ軽くなるような工夫が必要だと感じた．スプリントレビューは回数を追うごと指摘されることが少なくなり，サービスをもっと良くするための提案やアイデアが多く見られた．指摘の減少が良いことになるわけではない．しかし，初期のような根幹に関わる部分での指摘がなくなったことはよかったと感じる．また，普段はオンラインで行っていたデイリースクラムを対面で行えるようになったのはとても良かった．毎回対面で行えてはいなかったので，本来のスクラム開発では推奨されていないが，対面で活動できて有意義な活動になったと言える．1 年通して難しい判断や選択をする場面が多かったが，メンバーに助けられる場面が多くとても感謝している．今回の経験から今後の活動に活かしていきたいと考える．

（文責: 山内彩土）

6.3　佐々木紀明のふりかえり

プロジェクト活動を通してチームでの開発を経験し，多くのことを学んだ．私たちのプロジェクトでは，最初に全員がファシリテーターを経験した．1 週間ごとに 2 人ずつ交代で担当し，進行することの難しさを経験した．私は，一番初めにファシリテーターを経験したが初めてのオンライン会議だったこともうまく進行できなかった．ファシリテーターが 2 人だったことも原因と考えられるので，あらかじめどっちが進行をするかを決めておけばよかったと考える．前期の中盤ごろからファシリテーターを 1 人に決めたが，プロジェクトの中で発言をする人が固定化していたように感じた．私も，タイミングがつかめずあまり発言できなかった．プロジェクト内で発言がしやすい雰囲気づくりをしていれば良かったと感じる．また，反対意見がないかを聞かれることが多く賛成の場合に何も言わないという状況が多かったため，賛成の場合でもしっかりと発言するべきだったと感じる．前期に行った活動の中では特にフィールドワークを行ったことが深く印象に残っている．フィールドワークでは普段は気にかけない部分へ着目し，さまざまな問題点を見つけることができた．フィールドワークで見つけた疑問や問題点からアイデア出しにつなげることができたと感じる．アイデア出しの際には，さまざまな方法を知りそれぞれの長所や短所を学ぶことができた．しかし，アイデア出しの段階ではビーコンの知識が少なかったため，どのようなアイデアがビーコンに適しているのかわからず，サービスの決定に時間がかかってしまったと感じる．後期からはスク

ラム開発をした．Git や GitHub など初めて利用するツールもあり作業に手間どることもあったがスプリントを重ねることで作業に慣れ，スムーズに開発できたと感じる．開発を進めていく中で報告・連絡・相談の大切さを痛感した．自分のタスクの終了時にほかの人がどの作業をしているのかわからず，どのタスクに手をつけれよいかわからないことが何度かあった．さらに，自分も進捗報告をせずに開発を進めていてほかのメンバーを心配させたことなどもあった．この経験から報告・連絡・相談の大切さに気付き，終盤のスプリントではメンバーどうしでの報告回数が増え円滑に作業を進めることができるようになった．自分はスクラムマスターを担当し，デイリースクラムとスプリントレトロスペクティブ進行などを経験した．序盤のスプリントではあまり有意義なデイリースクラムができず，進捗報告会のようなものになってしまっていた．また，スプリントレトロスペクティブで決めたことがその次のスプリントに活かされないことも多かった．しかし，ほかグループのスクラムマスターとコミュニケーションを取りアドバイスをもらうことで良いところを吸収できた．デイリースクラムの後に詰まっているところなどを相談できる時間を設けたり，スプリントレトロスペクティブでは次回スプリントでやることを明確に決定して忘れないように取り組んだ．その結果，終盤のスプリントではとても有意義な時間を過ごせたように感じる．1 年間を振り返るともっと早く行動していればよかったなどの後悔や反省点が多く見つかった．だが，それを含めてもとても良い経験ができた．今回の経験を活かして今後の活動をしていきたい．

（文責: 佐々木紀明）

6.4　萩原啓道のふりかえり

今回のプロジェクト活動が初めての PBL，グループ開発の経験であった．そのため，多くのことを経験し，学ぶことができた．1 年の活動を通して前期では，PBL から学ぶことが多かった．そして，後期では，チーム開発から学ぶことが多かった．

まず，前期では，チームで活動することがほとんどであった．印象に残っている活動は，フィールドワークと提出物の管理である．初めに，フィールドワークは，アイデアを発見するため，函館の街で行った．注意深く，日常では見過ごすようなものにも注意深く観察することは新鮮な経験であった．本プロジェクトでは，ビーコンというテーマがあった為，ビーコンと掛け合わせられるものを考えた．テーマによって，選択肢が狭まり，アイデアを見つけやすくなったと考える．フィールドワークで気付いたことは，メンバー内で共有した．その後，アイデアを発散し，収束することを繰り返した．アイデアを共有することで，自分になかった着眼点へ気付くことができた．そして．収束することで，良いアイデアのみ残るため，アイデアの質が洗練された．発散と収束を繰り返すことで，アイデアの幅を広げることやアイデアをより深くできると感じた．前期では，本プロジェクトでの役割として提出物の管理を担当した．人をまとめ，責任を持つという体験があまりなかったので，たいへんな体験であった．たいへんだと感じた点は，日程の管理と取り組む方法の選択である．日程の管理では，締切を確認したうえで日程を立てた．しかし，余裕がなく締切に追われることとなった．時間の見積もりがうまくできなかったため，余裕を持てなかった．見積もり通り，うまくいかない可能性を考慮したうえで，見積もりをすることが必要であると考える．前期の活動を通じて振り返ると，自分の意見を持ち，共有することの難しさを感じた．本プロジェクトは PBL の活動であり，それぞれが問題に対して向き合い，チームで解決するものである．活動の中で，ディスカッションをする場面が多々あった．しかし，意見に対して肯定が多く，自分の意見を

うまく提案できなかった．初めての事が多く，経験の少なさから意見を出せないといったこともある．それ以上に，考える時間が足りていないと感じた．自分で考え，答えが出なくても考え続け，ほか人の意見を聞くことで，確かな思考力につながると考える．

そして，後期では，スクラムを採用した開発を経験した．開発を通して，大きく 4 つ学ぶことができた．1 つ目は，タスクを扱いやすい形にすることの大切さである．開発期間のうちの前半では，開発経験がなかったことから，各スプリントで設定するバックログのタスクに対するイメージが湧かず，細かくすることすら難しかった．しかし，開発を重ねるにつれ，タスクを細かくできた．タスクの大きさを取り組みやすい形にすることで，開発の効率が向上した．これは，タスクの細分化から，タスクに対するイメージがつかみやすくなる．つまずいた際には，問題が発生した箇所を早く発見できたためであると考える．2 つ目は，グループ開発においての報告，相談，連絡の大切さである．本開発で，作業をしている中で認識の違い，1 人で問題を抱えるケースが生じた．まず，認識の違いでは，開発を進めていく中で，認識のずれから実装した後に直すことが何度か生じた．グループ内での認識の擦り合わせが，不十分であったことから生じた問題である．あらかじめ，グループ内でマイルストーンやタスクを設定する際に，十分な話し合いをしたうえで適宜，連絡をすることの大切さを感じた．1 人で問題を抱えたケースでは，抱えた問題に対して手詰まりとなった際，1 人で抱え込み，開発を遅らせてしまった．メンバーを頼る事なく，抱え込むことはグループ開発としてふさわしくないと感じた．3 つ目は，ターゲットを考慮する大切さである．これは，スクラム開発にあるイベントのスプリントレビューとスプリントプランニングから，感じることが多かった．まず，スプリントレビューでは，スプリントの終わりにチーム内で，進捗やアイデアを発表し，意見をいただいた．グループ内では，役割が開発だった．そのため，スプリントレビューで実装した機能を発表する場面が多かった．発表する際に，開発期間の前半では，単なる進捗の報告となってしまう場面があった．イベントの意味を意識できていなかったと感じた．スプリントレビューは，プロダクトに対しての方向性を決めていくイベントであるため，単なる進捗報告の場面ではない．どんな体験や価値をスプリントで実装したかを発表することが適切であったと考える．スプリントプランニングでは，スプリント内でやるべきことを設定した．開発期間が約 2 ヵ月であった．そのため，開発する内容を取捨選択し，優先順位をつける必要があった．その中で，届けたい価値を考慮したうえで，ユーザーを意識することで，実装すべき機能が見えてきた．ユーザーを中心に考えることが，開発の中でも重要な要素であると感じた．4 つ目は，スクラムという手法の汎用性である．本開発では，スクラムを，開発の中で使用した．しかし，スクラムの考え方は，さまざまな用途や分野に適用できると考える．計画を達成するために，必要なものを書き出し，マイルストーンなどを設定しながら見直しを一定の周期で行う．このように，PDCA サイクルを繰り返す取り組み方は，問題解決において，柔軟に対応できる．今回得たスクラムの考え方を，今後の活動にも適用したいと考える．

1 年間を振り返ると，至らないところが多く，メンバーに多くの迷惑をかけてしまった．しかし，以上に挙げた多くのことを学ぶことができ，とても充実した 1 年間であったと感じる．今回の活動を 1 年間行えたことは，一緒に取り組んだメンバーや活動を支えていただいた先生方，TA，OB・OG のおかげである．本プロジェクトの先生方，TA，OB・OG，メンバーには感謝したい．そして，今回の経験を糧にして，今後の活動へ役立てたいと考える．

（文責: 萩原啓道）

第 7 章　まとめと展望

7.1　後期活動内容のまとめ

後期の活動では夏休みまでの活動で決定したサービスの機能を踏まえ開発を始めた．開発では，アジャイル開発手法の 1 つであるスクラムに挑戦した．スクラムはスプリントという期間を設けその中で計画，開発，ふりかえりを行いこれを繰り返していく手法である．他のグループと比べ人数の少ない 3 名での開発となったが，スクラムを行うことで完成形のイメージを確認しつつサービスを開発できた．スプリントレトロスペクティブでは，スプリントレビューで挙げられたサービスについてのレビュー内容をふり返った後，KTP 法を用いてスプリント期間中のふりかえりを行った．これにより，各メンバーの生活リズムを直すためにデイリースクラムの時間を変更した．デイリースクラムでは，スプリントゴールの進捗を確認しスプリントレビューまでの活動の計画をした．当初はデイリースクラムを正午に行っていたが，メンバーがデイリースクラムの時間まで寝ていることが多く，生活リズムを狂わせていたことがスプリントレトロスペクティブで確認され，朝開始することになった．サービスについては開発期間中に，お気に入り機能以外の機能を実装した．3 度目のスプリントレビューの際，当初予定していた飲食店や各種施設の店内 BGM として楽曲を共有させるというサービス内容を実現することが難しいとわかった．これを受けてサービスの内容を変更し，現在のサービス内容になった．サービス内容を変更する結果になってしまったのでサービス内容について十分に考える必要があったと言える．また，成果発表会では動画を含めた発表準備を行い，成果発表に臨んだ．

（文責: 山内彩土）

7.2　展望

今回実装したサービスには課題がいくつかある．1 つ目は，未実装の機能があることである．未実装の機能はお気に入り機能で，ユーザーがコミュニケーションを取るときに必要となる重要な機能である．この機能がないと，会話以外でユーザーどうしがコミュニケーションを取ることができないため，それぞれで音楽を共有しあっているだけになってしまう．2 つ目は，どこでサービスが利用できるのか分からないことである．ビーコン範囲内にいる時にのみこのサービスを利用できるが，現状ユーザー側は設置されているビーコンの場所は分からない．サービス内でどこにビーコンが設置されているかわかるようにするか，ビーコンが設置されている施設などにサービスが使えることを伝える広告のようなものを掲示する必要があると考える．また，現在実装されている機能にも課題がある．サービスを使うためには Spotify のプレミアム会員になっている必要があり，現時点では多くのユーザーにとって使うことが難しい仕様になってしまっている．これらを改善してユーザーが利用しやすいようなサービスを開発していきたい．

（文責: 山内彩土）

参考文献

[12] **来函観光入込客数推計**. https://www.city.hakodate.hokkaido.jp/docs/2015062500021/files/R02irikomi.pdf. (Accessed on 01/07/2022).

[13] **北海道新型コロナウイルス感染症対策本部**. https://www.pref.hokkaido.lg.jp/covid-19/G01/f50/. (Accessed on 01/07/2022).

[14] *SwiftUI Tutorials | Apple Developer Documentation*. (Accessed on 01/19/2022).

[15] *What is Slack? | Slack*. (Accessed on 01/19/2022).

[16] **[初心者向け]** *Discord* **(ディスコード) とは | 使い方から注意点まで解説** | *niftyIT* **小ネタ帳**. (Accessed on 01/19/2022).

[17] *Zoom Meetings | Zoom*. (Accessed on 01/19/2022).

[18] *esa* - **自律的なチームのための情報共有サービス**. (Accessed on 01/19/2022).

[19] *Miro(***オンラインホワイトボード***) | aslead |* **野村総合研究所** *(NRI)*. (Accessed on 01/19/2022).

[20] **無料で使える** *UI* **デザインツール「***Figma***」を使ってみた |** *Solution* - **ソリューション | 株式会社イージェーワークス**. (Accessed on 01/19/2022).

[21] *Adobe Illustrator |* **グラフィックデザインソフト【アドビ公式】**. (Accessed on 01/19/2022).

[22] **【初心者向け】** *Git* **とは何なのか。基本用語やその仕組みをまとめています。| ワードプレステーマ** *TCD*. (Accessed on 01/19/2022).

[23] *GitHub | GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside millions of other developers.* (Accessed on 01/19/2022).

第 IV 部

FLAT について

第 1 章　背景

1.1　背景

近年，スマートフォンアプリケーションを用いた位置情報共有サービスとして，GPS を利用したものが多くリリースされている．位置情報共有サービスの使用用途として，「友だちや親と位置情報を共有」「スマートフォンを紛失した際に見つける」などが挙げられる．いつでもスマートフォンで連絡を取れるようになった今，時間と場所をあらかじめ決めて待ち合わせするのではなく「LINE」や「Instagram」で連絡をしながら待ち合わせをする人たちが増えてきている．そこで，位置情報共有サービスを用いることにより map 上で相手の居場所を確認できるため，ほとんど連絡を取ることがなく待ち合わせをできる．しかし，既存の GPS を用いたサービスでは屋内の詳細な位置情報までは測位できない．そのため，屋内での待ち合わせでは連絡を取る必要が発生してしまう．屋内で位置情報を共有するサービスもあるが，私たち未来大生が学内で使用できるサービスは現状存在しない．

本サービスは，未来大生に向けて，ビーコンを用いた未来大内限定で屋内での位置情報の共有を提供するサービスである．

（文責: 大巻佑太）

1.2　課題

現在，未来大内で詳細な位置情報を共有できるサービスはないため，学内で友だちのもとを連絡をとらずに訪ねるのは難しい．その課題を解決するために，未来大内限定で位置情報を共有できるようにし，現状よりも気軽に友だちのもとを訪ねられるようにする．

（文責: 大巻佑太）

第 2 章　目的

本サービスの目的は，未来大内に多数設置されているビーコンを用いて屋内の詳細な位置情報を共有できるサービスを開発し，提供することである．これにより，屋内でも連絡を取ることなく友だちの元を訪ねることができる．

（文責: 大巻佑太）

第 3 章　サービスの概要

3.1　サービスの概要

本サービスは，未来大内限定で友だちと位置情報を共有できるサービスである．未来大内に多数設置されているビーコンを用いて，ビーコンの範囲内にいる友だちの居場所を知ることができる．

本サービスを利用することによって、友だちを訪ねる際に連絡をする必要がなくなり，

今までよりも気軽に友だちを訪ねることが可能となる．

（文責: 大巻佑太）

3.2　ユーザーストーリー

本サービスは，未来大生を対象としている．ユーザーは自身の友だちのユーザー名を検索し，友だち申請を送る．友だちが送られてきた申請を承認すると，お互いの友だち一覧に友だちとして追加される．友だち一覧に追加されたユーザー名の右側に居場所が表示されているので，そこから相手の居場所を確認する．友だちの元を訪ねる．

以上が，本サービスのユーザーストーリーである．

（文責: 大巻佑太）

第 4 章　アプリケーションの開発

4.1　機能

本サービスの機能は「アカウント作成機能」「ログイン・ログアウト機能」「友だち検索機能」「友だち追加機能」「友だち一覧表示機能」の 5 つに分けられる．

（文責: 大巻佑太）

4.1.1　アカウント作成機能

「アカウント作成機能」（図 4.1）は，ユーザーを識別するための機能である．アカウント作成の際に，ユーザーにニックネームとパスワードを決めてもらう．ニックネームは重複不可能であり，変更は可能である．パスワードは半角英数 8 文字以上としている．

図 4.1　アカウント作成画面

（文責: 大巻佑太）

4.1.2　ログイン・ログアウト機能

「ログイン機能」（図 4.2）は，ユーザーが別の端末でも本サービスを利用できるようにするための機能である．これにより，スマートフォンを変更した際にも，継続して本サービスを利用することが可能となる．

「ログアウト機能」はアプリケーションを実際にリリースするにあたって必要な機能である．ログアウトをする際に，確認のためのダイアログが表示されるため，誤ってログアウトすることがな

いようになっている．

図 4.2　ログイン画面

（文責: 大巻佑太）

4.1.3　友だち検索機能

「友だち検索機能」（図 4.3）は，友だちのユーザー名を検索し，申請を送ることができる機能である．検索欄に友だちのユーザー名を入力し，検索すると該当のユーザーが表示される．このとき，申請ボタンも一緒に表示されるため，タップすると申請を送ることができる．すでに申請済みの場合は「承認待ち」，すでに友だちの場合は「すでに友だち」と表示される（図 4.4）．

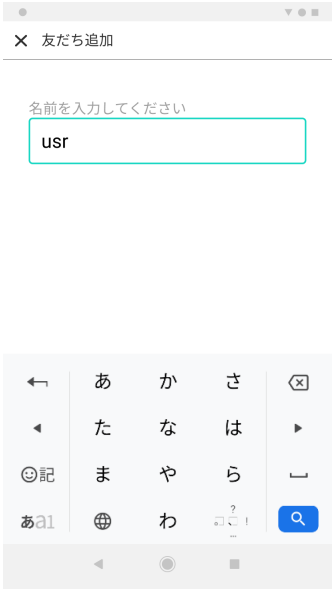

図 4.3　友だち検索画面

図 4.4　検索結果

（文責: 大巻佑太）

4.1.4　友だち追加機能

「友だち追加機能」（図 4.5）は，送られた申請に対して承認または拒否を選択できる機能である．拒否を選択時，本当に拒否するかを確認するダイアログが表示されるため，誤って拒否してしまうことがないようになっている（図 4.6）．自身が申請を送り承認された場合は，承認された時点で友だちとして追加される．

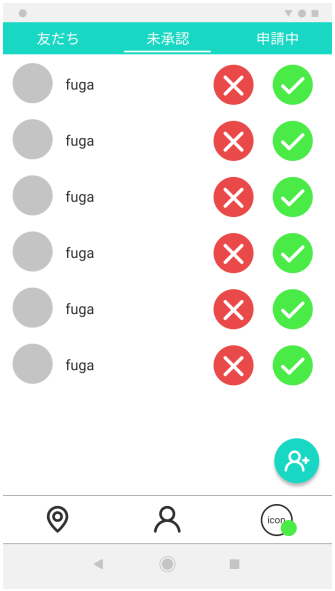

図 4.5　友だち追加画面

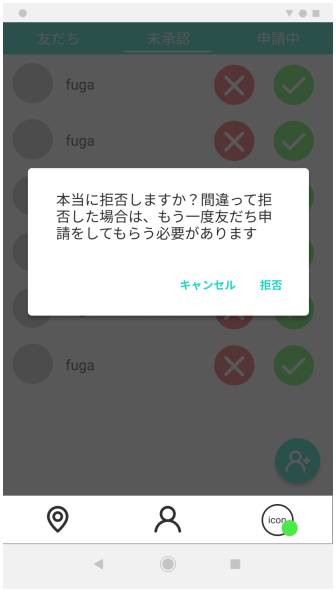

図 4.6　ダイアログ

（文責: 大巻佑太）

4.1.5　友だち一覧表示機能

「友だち一覧表示機能」（図 4.7）は，友だちになった相手を一覧として表示する機能である．友だち一覧とは別に，未承認の友だち一覧も別のタブとして表示される．未承認の友だち一覧では，自身に友だち申請を送ってきた相手を一覧で確認できる．友だち一覧では，友だちの居場所を確認できるようになっており，友だちが未来大内にいる際，居場所が表示される．未承認の友だち一覧では，まだ承認していないため居場所は表示されない．

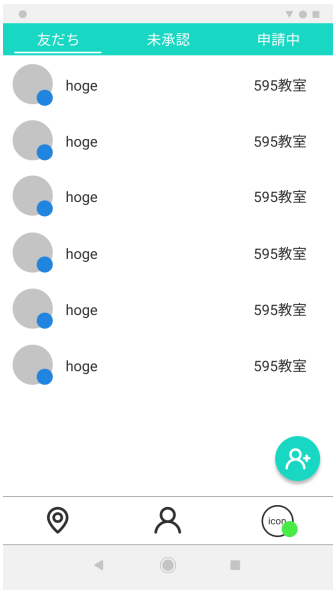

図 4.7　友だち一覧表示画面

（文責: 大巻佑太）

4.2　開発手法

4.2.1　使ったツール・サービスについて

本サービスは，サービス案の決定や開発にさまざまなツールを使った．以下にチーム開発を進めるために用いたツールやサービスについて記述する．

（文責: 小田嶋亜美）

Slack

Slack とは，ブラウザや専用のアプリケーションから使用できるビジネスチャットツールであり，連絡用ツールとしてさまざまな企業で使用されている [24]．プロジェクト全体やチーム内の連絡，教員や TA との連絡に用いた．Slack はチャンネルを登録している人全員がメッセージのやりとりを見ることができ，チーム外の活動や連絡なども随時確認できる．また，気軽にメッセージを送ることができ，ダイレクトメールで個人的にメッセージを送ることもできるため使用した．

Git

Git とは，プログラムのソースコードなどの変更履歴を記録・追跡するための分散型バージョン管理システムである [25]．プログラムのソースコードを共有，追跡するために用いた．Git を使い，チームメンバーのソースコードを確認し，変更点を更新することによって，開発作業に役立てた．

GitHub

GitHub とは，ソースコードをホスティングできるソフトウェア開発のプラットフォームであり [26]，他の開発者と一緒にソースコードのレビューをしたり，プロジェクトの管理をしながら，ソ

フトウェアの開発ができる．Git のリポジトリを保存し，開発したソースコードを公開，レビューするために使用した．また，GitHub 上でタスクの管理も行い，チーム内のタスクの進行状況を確認できた．

esa

esa とは，自律的なチームビルディングのためのドキュメント共有サービスである [27]．日々の活動を議事録として書きとめるために用いた．esa に書き残すことによって前回の活動内容を確認できた．また，esa は同時に編集できるため，活動時間中に複数のメンバーが協力し合って，書きとめることができた．

Miro

Miro とは，オンライン上で共同編集ができるホワイトボードプラットフォームである [28]．本サービスのユーザーストーリーを作成するために用いた．Miro 上に表を作り，プロダクトバックログを作成した．

Discord

Discord とは，Windows・macOS・Linux・Android・iOS などで動作する，インスタントメッセージ・ビデオ通話・音声通話ができるソフトウェアである [29]．朝会，夜会の作業通話に用いた．朝会では，プロダクトオーナーが GitHub の画面を共有し，タスクの進行状況を確認した．夜会では，通話しながらチームメンバーどうしで開発上の困っていることを相談し，モブプログラミングを行って問題を解決するために役立てた．

Figma

Figma とは，ブラウザ上で簡単に使用できるデザインツールである [30]．サービスのデザイン，ロゴの作成に用いた．サービスのデザインでは画面遷移のシミュレーションやカラーパレットの作成をし，モバイルアプリケーションの開発に役立てた．ロゴ作成では，ロゴ案を複数作成し，ロゴのイメージやデザインを確認して最終決定に役立てた．

Zoom

Zoom とは，PC やスマートフォンなどのデバイスを使用して，オンラインでセミナーや会議を開催するためのアプリケーションである [31]．毎週 2 回行われるシステム情報科学実習の時間にプロジェクトメンバー全員で集まるために使用した．ブレイクアウトルームを作成し，チーム内での活動に役立てたほか，チャット機能やリアクション機能を活用して，メンバー間のコミュニケーションを取ることができた．

Android Studio

Android Studio とは，Google が提供する統合型開発環境である [32]．本サービスの Android アプリケーションの開発に用いた．

Visual Studio Code

Visual Studio Code とは，Microsoft 社が開発しているエディターであり [33]，多くの言語に対応できるほか，拡張機能と呼ばれるしくみで機能を追加できる．本サービスのサーバーサイドアプリケーション，報告書の作成に用いた．

Xcode

Xcode とは，Apple 社が提供する統合型開発環境である [34]．本サービスの iOS アプリケーションの開発に用いた．

Adobe Illustrator

Adobe Illustrator とは，Adobe 社が提供しているグラフィックデザインツールであり [35]，テキストと画像を組み合わせたレイアウトの作成やデザイン，線や図形を組み合わせたイラストの作成ができる．ポスターの制作，本サービスのロゴ制作に用いた．ポスター制作では，ポスターをクラウドで共有し，進行具合の確認や修正をメンバーどうしで同時に行うことができた．ロゴ作成では，Figma でロゴのデザインを作成し，Adobe Illustrator でアプリケーションアイコン用にロゴを清書した．

Google スライド

Google スライドとは，ブラウザ上でプレゼンテーション資料を作成できるサービスであり，同時に共同編集できる [36]．プレゼンテーション，発表会の資料の作成に使用した．

Google ドキュメント

Google ドキュメントとは，ブラウザ上でテキスト文書の作成，編集できるサービスであり，作成したファイルを共有，同時に共同編集できる [37]．プレゼンテーション，発表会用の台本の作成に使用した．

（文責: 小田嶋亜美）

4.2.2 開発手法について

アジャイル

本サービスの開発では，ソフトウェア開発手法の 1 つであるアジャイルを導入した．チーム発足時は，新型コロナウィルス感染症対策のために作られた学内向け位置情報記録アプリケーション「LATTE」の改修に加えて新規サービスの開発をしようとしており，開発期間が約 2 ヵ月しかないことを考慮して迅速に開発するためにこの手法を採用した．開発するサービスの詳細を決めていくうえで「LATTE」の改修はせずに新規サービスの開発のみをすることとなったが，開発期間が変わらず短いことや，メンバーがすでにアジャイル開発手法の 1 つであるスクラムについて勉強していたことなどから，開発手法は変更せずに本サービスを開発した．

（文責: 柿本翔大）

スクラム

チーム構成　本サービスの開発では，アジャイル開発手法の 1 つであるスクラムを採用し，4 人のメンバーでチームを構成した．チームはプロダクトオーナー，スクラムマスター，開発メンバーの 3 つのロールで構成されており，ロールの決定は，メンバーがやりたいロールを宣言する立候補制にて行った．プロダクトオーナーは立候補した者が 1 人だったため希望通りに決まったが，スクラムマスターを希望した者がおらず，開発メンバーを希望した者が 3 人となった．そこで，開発メンバーを希望した 3 人のうち 1 人が，どうしても開発をしたいため開発メンバーを兼任する形でならスクラムマスターを引き受けると発言したことでメンバー全員のロールが決定した．したがって，私たちのチームは専任のプロダクトオーナー 1 人，開発メンバーを兼任するスクラムマスター 1 人，専任の開発メンバー 2 人から構成される．

（文責: 柿本翔大）

各イベントをどのように行ったか　最初に，1 スプリントの期間については固定せず，開始を木曜日，終了を水曜日として各スプリントのスプリントプランニングの際に期間を 1 週間にするか 2 週間にするか決定した．1 スプリントの期間を固定しなかった理由は，開発メンバーの能力を考慮した結果 1 週間では終わらないが，それ以上細分化できないバックログが存在したためである．事前に 1 週間では終わらないことがわかっている状態でスプリントの期間を 1 週間と定めることは無駄な行為だと判断し，スプリントの期間はスプリントプランニングの際に決定することとした．

スプリント期間中は毎日デイリースクラムを行った．デイリースクラムのために開発メンバーには，Slack に作成したチャンネルで毎朝 7 時に投稿されるリマインドに対して「 昨日やったこと」「今日やること」「スプリントゴールを達成するうえでの問題」の 3 つの項目について返信してもらった．平日は 8 時 30 分に Discord で通話をつないで返信した内容について発表した．通話にはプロダクトオーナーも参加し，タスクを管理している画面を共有することで進捗確認を兼ねた．発表が終わり次第 9 時まではメンバー間でコミュニケーションを取る機会として雑談の時間を設けた．休日は平日よりもゆっくり休みたいというメンバーの意見から，12 時までに 3 つの項目について返信をすることで，非同期型のデイリースクラムを行った．

スプリントレビューは水曜日のプロジェクト学習の時間に行った．最大 30 分で発表，デモンストレーション，質疑応答を行い，質疑応答の際に得られた意見を次回以降のスプリントに活かした．また，円滑な発表およびデモンストレーションにより質疑応答の時間を多く確保できるように，必要に応じてデモンストレーションの動画をあらかじめ撮っておくなどの工夫をした．

スプリントレトロスペクティブは水曜日の 18 時 30 分から 20 時の 1 時間 30 分で行った．最初にスプリント内で起きたことについての共通認識を得るために，スプリント期間中に何をしたのかや，それで何を感じたのかを自由に書き起こした．書き起こしたものはすべて読み上げ，一言コメントをつけた．次に，次回のスプリントをより良くするために何をすべきかを考えるため，KPT 法を行った．KPT 法は「Keep（今回のスプリントで行った良かったこと，継続すべきこと)」「Problem（今回のスプリントで起きた問題や課題)」「Try（次回のスプリントで新たに実践すること，問題や課題の解決策)」を順番に考えるふりかえり手法である．メンバー間で具体的な共通認識を得るため，Try の案を出す際にはその Try を実行した結果どのような状態になるのかを含めて考案した．その後，多数出た Try の案から次回のスプリントで実際に行うものを投票により決

定した．投票では 1 人 3 票を次回のスプリントで行うべきだと判断したものに投じ，得票数の多いものから順に 2 つから 4 つほどを選んだ．最後に，スプリントレトロスペクティブ自体のふりかえりを行い，そこで出た意見を次回以降のスプリントレトロスペクティブにおける時間管理や進行方法などに活かした．

（文責: 柿本翔大）

独自の取り組み　私たちのチームでは，第 1 スプリントを開始する前に準備期間として第 0 スプリントを設けた．第 0 スプリントでは，開発環境の構築や必要な技術の習得，UML（Unified Modeling Language）を用いた設計，モックサーバーの構築，スプリントの流れの確認など，事前に行うことで開発を円滑に進める一助となると判断したものを行った．

ほかにも，私たちのチーム独自の取り組みとしてオンラインでのモブプログラミングを行った．モブプログラミングとは，ナビゲーターと呼ばれる複数人の人たちが指示を出し，それを参考にしてドライバーと呼ばれる 1 人が実際にコードを書くプログラミング手法である．今回はオンラインで行ったため，ドライバーはコードを書いている画面を共有し，ナビゲーターはそれを見て指示出し等を行った．モブプログラミングを用いることでコーディングとレビューを同時に行うことができ，ドライバーはナビゲーターからコーディングについて学ぶことができる．私たちはこのモブプログラミングをメンバーが行いたいと発言した際に不定期的に行ったほか，第 4 スプリント以降のスプリントで土曜日の 21 時から 2 時間行うと決定し，定期的に行った．モブプログラミングを定期的に行う時間を土曜日の 21 時に決定したのには理由がある．休日のデイリースクラムを非同期型にした結果，週末に発生した問題をうまく文章に起こせず，メンバーが抱えている問題の重大さを月曜日のデイリースクラムで初めて知るといったことが多発したためだ．休日のちょうど中間である土曜日の夜に通話をつなぐことで，仮に Slack で問題を報告できていなかったとしても問題の発生から時間を置かずにチームでしっかりと問題を把握し，対処するという目的をもってこの時間に決定した．また，土曜日の 21 時から行われるモブプログラミングをその開催される時間から夜会と称し，23 時以降は雑談の場としてチームメンバーの仲を深めるための時間を設けた．

（文責: 柿本翔大）

4.3　開発にあたっての決め事

本サービスを開発するにあたって，リポジトリを効率的かつ安全に運用するために以下のことを定めた．

Git Flow

Git Flow とは，2010 年に Vincent Driessen によって提唱されたブランチモデル [38] の通称および，それを実現するためのコマンド [39] である．表 4.1 では GitHub のデフォルトブランチ名が，master ブランチは main ブランチへと名前を変更された [40] ためそれに追従している．本来は，これに加えて release ブランチというものがあるが，release ブランチはコードの規模に対し冗長と判断したため今回の開発で採用していない．そのため説明を省略する．

本サービスの開発では，各リポジトリのソースコードを GitHub 上で管理していた．個々人が規約を設けず自由にリポジトリを運用した場合，1 つのブランチの粒度が粗くなり，プロダクトバッ

ブランチ	役割
main	リリース用ブランチ．タグを打ち，実際に動くものを配置する．
develop	開発用ブランチ．ここに機能を追加する．リリースの準備ができたら main にマージする．
feature branches	機能追加用ブランチ. ここで機能を実装し, develop にマージする．
hotfixes branches	リリース後の致命的なバグなどを修正するブランチ．

表 4.1　Git Flow における各ブランチとその役割

クログアイテム外の作業をする恐れがある．main ブランチに直接コミットをすることで，大量のコンフリクトが発生するなどの懸念があった．Git Flow では，ある feature ブランチ 1 つにつき 1 つの機能を実装する，develop ブランチに実装した機能をマージする，などの制約がある．これらの制約によって，上記の問題が解消されるため，Git Flow を導入することを決めた．

Git Flow を導入してよかったと感じたことは以下の点である．

- ブランチ操作を定型化できたため，自分がどのブランチにいるかを間違うことが少なかった
- 1 つの feature ブランチが肥大化することを防ぐことができた
- 1 つのリポジトリを複数人で作業することがあったが，マージする際に，ほとんどコンフリクトすることがなかった
- Git Flow コマンドを用いることで，必ず develop ブランチから feature ブランチを切ることができる．これにより，分岐元を間違うことを減らせた

（文責: 山本竜生）

4.3.1　コーディング規約

コーディング規約は以下の 3 点のみ定めた．チームでほかに問題が発生したらその都度追加するという話をし，それに合意した．厳しく縛るよりも，ある程度の裁量をもたせることで，すばやく開発できると考えたからである．最後まで大きな問題は発生せず，規約が増えることはなかった．

- 変数の中身と意味が違う変数名を付けない
- 関数の中身と意味が違う関数名を付けない
- 1 つの関数には 1 つの機能のみをもたせる

変数名についての規約は，後からコードを読み返すときに，コードを追いかけやすくするためのものである．不用意に短い変数名や，変数の中身と大きく乖離した変数名を使用すると，コードが大きくなったときに可読性が落ちるからである．可読性の低下は，バグの混入につながるため特に気を付けた．関数名についての規約は，コードの局所化を目的として定めた．関数と書いてあるが，ほぼ同様の機能を持つメソッドにもこの規約を適用した．変数名と同じく，実際の処理と大きく乖離した関数名をつけると可読性が落ちる．その関数の責務があいまいになるなどの問題点がある．

適切な関数名が付けられていることで，詳しい処理の中身を見ることなく，コードを読み進めることができるという利点がある．1 つの関数に 1 つの機能という規約は，関数名の規約と同様にコードの局所化を目的として定めた．関数についての 2 つの規約によって，適切な関数名に適切な処理が実装されることを期待できる．

（文責: 山本竜生）

Pull Request 時のレビュー

フロントエンド開発では，feature ブランチから develop ブランチに Pull Requset を発行した際に，必ずレビューをした．バックエンド開発では，用意したテストケースを通すことを条件として，Pull Request を承認したため，レビューをしていない．本節ではフロントエンド開発に焦点を合わせる．

レビューは，4.3.1 節の規約を中心に行ったが，動くことを第一とし，軽微な修正についてはコメントのみ．または，Issue を立てる程度にとどめることもあった．これは，レビューはコードの品質を高めるものではなく，最低限を保証するものであるというチームでの合意にもとづく．

レビュアーは，自分の PC でアプリケーションをビルドし，実際に動くことを確認した．加えて，GitHub 上でコードレビューをし，第 5 スプリントからは UI テストを導入し，UI テストに合格することをレビューの要件に含めた．これによって，開発者とレビュアーの間に共通認識ができた．共通認識ができることによって，勘違いや実装漏れが減り，よりよいレビューを行えた．

作業者が誤って，develop ブランチに変更を取り込んでしまったことがあったため，develop ブランチに対して，GitHub の Branch protection rule[41] 機能を適用した．Branch protection rule を適用することで，レビューされていない変更が取り込まれることや develop ブランチに対して直接プッシュすることを防ぐことができる．

（文責: 山本竜生）

4.4 効率的な開発のために行ったこと

4.4.1 メンバー間のコミュニケーション

効率的な開発のためにメンバー間のコミュニケーションをさまざまなツールやスクラムのイベントで重点的に行った．

（文責: 小田嶋亜美）

Slack

Slack のチャンネルで適宜問題点を報告して，チームメンバーが協力して解決策を見つけることができた．また，メンバーどうしで気軽に質問し合い開発の役に立てた．

朝会

開発期間中は朝会を行い，タスクの進捗を確認した．朝会を Discord で通話をし，同期型で行ったことにより，チームメンバーが口頭でタスクの進捗を聞くことができた．また，朝会自体は軽く行い，その後の雑談で現在抱えている問題を共有した．実際に話すことで，根本的な原因をその場ですぐに究明できた．

夜会

第 3 スプリントのスプリントレトロスペクティブで夜会としてモブプログラミングの実施を決定した．夜会を実施した理由は，仮に Slack で問題を言えていなくてもチームで具体的な問題点を把握できるという意見が出たためである．夜会を行うことで，通話をしながら開発作業をする中で気軽に質問ができ，チーム内のコミュニケーションをより活発に取ることができた．

スプリントレトロスペクティブ

スプリントレトロスペクティブでは，雑談を多めにして，チームメンバーが話しやすい雰囲気作りを心がけた．発言回数が少ないメンバーを指名して先に話すよう促し，その後の発言回数の改善をみることができた．

（文責: 小田嶋亜美）

4.4.2 タスク管理

GitHub Projects Beta という機能を使ってタスクの管理をした．

（文責: 小田嶋亜美）

GitHub Projects Beta

GitHub Projects Beta とは，GitHub の Project でタスクの管理ができる機能である．チーム全体のタスクを一覧で見られるほかに，各アプリケーション内のタスク，個人のタスクも確認できる．ラベルをつけて，どのタスクなのかをわかりやすく表示させることもできる．また，各タスクのステータスをタスク終了時に変更すると，タスクの進捗状況が明確になり，どのタスクが問題になっているのかを確認できる．

（文責: 小田嶋亜美）

第 5 章　実装

5.1　システム構成

本サービスのシステム構成図を図 5.1 に示す．本サービスは，モバイルアプリケーション（Frontend）とサーバーサイドアプリケーション（Backend）から構成されている．モバイルアプリケーションがビーコンから電波を定期的に受信し，モバイルアプリケーションの最も近いビーコンの情報をサーバーサイドへと送信する．それぞれの詳細は次節以降にある．

（文責: 山本竜生）

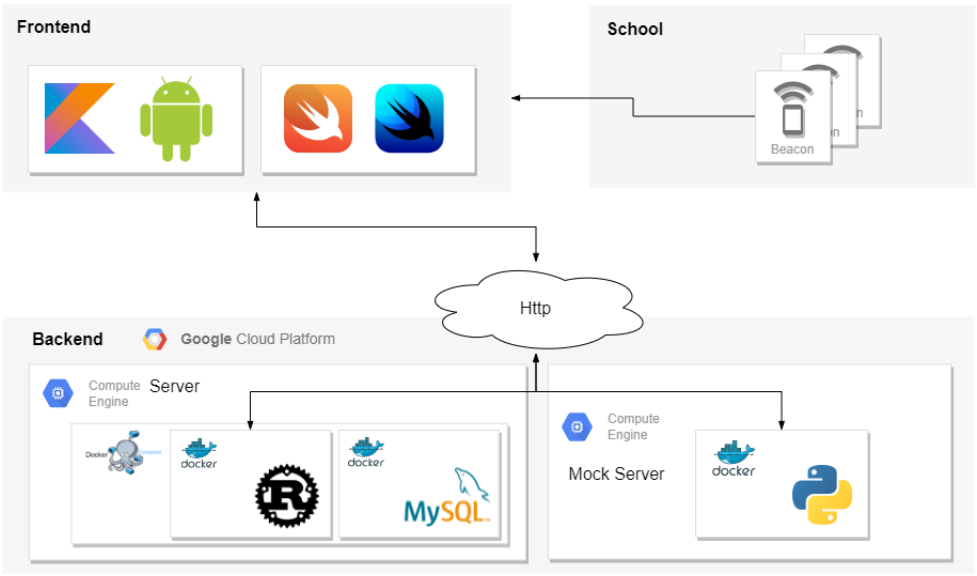

図 5.1　FLAT のシステム構成図

5.2　モバイルアプリケーション

5.2.1　iOS

開発環境

本サービスの iOS アプリケーションでは，MacBookPro を使って開発した．iOS アプリケーションを開発するために Apple 社がリリースしている開発者向けツールである Xcode[34] を用いて開発した．使用した Xcode のバージョンは Xcode13.1 である．

言語，フレームワーク

使用した言語は iOS，Mac，AppleTV，AppleWatch 向けのアプリケーションを開発するために Apple 社が作った Swift 言語 [42] を使って開発した．使用した Swift 言語のバージョンは Swift5.5.1 である．使用したフレームワークは SwiftUI である．SwiftUI を使用した理由は 2 つあ

る．第一の理由として，UI の実装が Storyboard よりも簡単に行えるからである．第二の理由として，コーディングをしながら実際のプログラムの動きを確認できるからである．SwiftUI は対応する環境が iOS 13 以降でなくてはならず，iOS 13 以降にも対応できるようにコードの書き方を工夫した．

利用したライブラリ

使用したライブラリは CoreBluetooth，CoreLocation と UIKit である．CoreBluetooth ではビーコンの検知のために使用し，ビーコンを検知するスマートフォン（セントラル）と検知したビーコン（ペリフェラル）を判別するために使用した．CoreLocation では，検知した複数のビーコンの RSSI 値を測り，最大であるビーコンの場所を検知するために使用した．UIKit では，アプリケーションの構築に必要なオブジェクトを作成するために使用した．

アカウント登録機能

iOS アプリケーションのアカウント登録機能では，ユーザーがアプリケーション上でのニックネームとパスワードを決めることができる．ニックネームは 10 文字以内に指定し，条件を満たしていなければエラーメッセージを表示させるようにした．パスワードは 2 回入力させるようにし，どちらも同じものではない時，エラーメッセージを表示させるようにした．

ログイン・ログアウト機能

iOS アプリケーションのログイン・ログアウト機能では，ユーザーがアカウント登録で決めたニックネームとパスワードでログインできる．また，ログアウトをした際，アプリケーションを再度インストールした時に同じニックネームでログインできる．

友だち検索機能

iOS アプリケーションの友だち検索機能では，ユーザーがアカウント登録時に作成したニックネームであるユーザーの名前を検索し，その後結果を表示させた．ニックネームは 10 文字以内でバリデーションチェックを行い，該当しなければエラーメッセージを表示するようにした．名前検索した結果の友だちに友だち申請ができる．検索時に，すでに友だちの場合や申請を送っていた場合もボタンの表示が変わるようにした．

友だち追加機能

iOS アプリケーションの友だち追加機能では，友だち申請が来た友だちを承認，拒否できることがボタンで選択できる．承認をすれば，相互に友だちとなり，一覧画面に追加される．拒否をした場合はダイアログを表示させ，再度確認をとるようになっている．

友だち一覧表示機能

iOS アプリケーションの友だち一覧機能では，友だち一覧画面で友だちを表示できる．画面では，複数の友だちをリストで表示している．また，画面右上のタブで友だちと未承認の友だちを選択でき，ユーザーが承認をしていない友だちも画面で確認できる．

（文責: 小田嶋亜美）

5.2.2 Android

開発環境

本サービスの Android アプリケーションでは，Google が提供する統合型開発環境である Android Studio - Arctic Fox | 2020.3.1 Patch 3 を用いて開発した．Android Studio のバージョンは開発開始時の最新バージョンが Arctic Fox | 2020.3.1 Patch 2 であったが，直後に Arctic Fox | 2020.3.1 Patch 3 が利用可能となったため，アップデートを実施した．ビルドツールは Gradle のバージョン 7.0.3 を使用し，プログラミング言語は Kotlin を使用した．Android 開発では Java が使用されることもあるが，Kotlin には Java との相互運用性があること，言語機能や型システムにより Java よりも安全で簡潔なコードが書けることなどの理由で今回は Kotlin を使用した．targetSDK バージョンは，開発開始時に最新であった Android 11（API Level 30）とし，最小 SDK バージョンは Android 8（API Level 26）とした．Android Studio では新規プロジェクトを作成する際にバージョン別シェアが表示される [43]．表示されるバージョン別シェアでは Android 8 以上が 82.7% を占めており，実際にユーザーにアプリケーションを利用してもらうには十分な数値だと判断したため，最小 SDK バージョンを Android 8（API Level 26）とした．

（文責: 柿本翔大）

アーキテクチャ

本サービスの Android アプリケーション開発では，MVVM アーキテクチャ +Repository パターンを適用した．MVVM とは Model View ViewModel の略称で，この 3 つのコンポーネントに責務を分離することでアプリケーションを管理しやすくしたり，堅牢性を高めるアーキテクチャパターンの 1 つである．View は UI を管理し，ViewModel は View が UI を表示するために必要なデータを取得して適切に処理し，Model はデータを保存する役割がある．通常の MVVM アーキテクチャでは参照できるコンポーネントが View の場合は ViewModel，ViewModel の場合は Model のみとなっているが，Repository パターンを適用することで ViewModel が Repository を参照し，Repository が Model を参照する形となる．また，ViewModel の役割がデータの処理を行うことのみに変化し，Repository が Model や Web サーバーなどからデータを取得することとなる．したがって，通常の MVVM アーキテクチャよりも各コンポーネントの責務を細かく分離できるようになり，アプリケーションの保守がより容易となる．

MVVM アーキテクチャ +Repository パターンには上記の利点があり，開発開始時に Google が推奨していたアーキテクチャであったことや，このアーキテクチャに適したコンポーネントやライブラリが提供されていることから，今回は MVVM アーキテクチャ +Repository パターンを適用した．しかし，本格的な Android アプリケーション開発をした経験がなく Android アプリケーション開発と学習を同時に行っていたことや，きれいなアーキテクチャよりも実装の速さを優先すると事前にメンバーとの話し合いで決めていたことなどから，アーキテクチャにのっとった実装方法がわからず MVVM アーキテクチャ +Repository パターンとは言いにくくなってしまった部分があるため，今後すべての機能を実装し終えた後にリファクタリングを行い，保守しやすいコードにする予定である．

（文責: 柿本翔大）

ビーコン検知機能

本サービスでのビーコン検知機能は AltBeacon というライブラリを用いて実装した．AltBeacon は Android で iOS と同じようにビーコンを検出，使用できるようにするライブラリである [44]．このライブラリを使用し，BeaconParser というクラスで iBeacon を識別できるように設定することで未来大内に設置してある iBeacon が検出できるようになる．しかし，そのままではすべての iBeacon を検出してしまい位置情報の表示に支障があったため，検出した iBeacon に対し Major 値と Minor 値を用いて絞り込みを行うことで，対象としている iBeacon かどうかを判別した．

本サービスでは，targetSDK バージョンを Android 11，最小 SDK バージョンを Android 8 としたが，Android 10 以降で権限に関する制限が強くなったため，ビーコンを検出するにあたってユーザーに要求する位置情報に関する権限をユーザーが使用する Android のバージョンによって分ける必要があった．Android 8 と Android 9 においては，位置情報へのアクセス権限のみをユーザーにリクエストすれば良いが，Android 10 においてはそれに加えて同時にバックグラウンドでの位置情報へのアクセス権限をリクエストしなければならない．Android 11 においては，必要な権限は Android 10 と同様だが，位置情報へのアクセス権限とバックグラウンドでの位置情報へのアクセス権限を同時にリクエストしてはいけないという制限があるため，ユーザーに対しそれらの権限を順番にリクエストするようにダイアログを実装した．また，ビーコンの検出をサービスとして実装し，サービスの実行中にはステータスバーに常に通知を表示することでバックグラウンドでもビーコンを検出できるようにした．

（文責: 柿本翔大）

アカウント作成画面

本サービスは，アプリケーション起動時にアカウントを新規作成するかログインするかを選べるようになっている．アカウントを作成する場合ユーザーはニックネームの入力，パスワードの入力，パスワードの再入力をした後，ボタンを押下することで入力情報をサーバーに送信し，問題がなければ入力したニックネームとパスワードでアカウントが作成され，サーバーで生成される ID が割り当てられる．端末には ID とニックネームを保存し，これらの情報を用いることでユーザーの識別を可能とする．ユーザーがボタンを押下した際，ニックネームが空の場合や 2 つのパスワードが一致しない場合，パスワードに半角英数字以外が使われている場合，指定した文字数に達しない場合などは入力情報をサーバーへ送信する前にエラー表示をしてユーザーに再入力を呼びかける．入力に誤りがなく，入力情報を正常にサーバーに送信できたとしても，既存のユーザーとニックネームが重複した場合はアカウントの作成は行わず，ニックネームを変更するようユーザーに促す．このようにエラーを切り分け，端末側でエラーチェックを行えるものはすべて端末でエラーチェックを行うことで，サーバーと余計な通信をしなくても済むようにした．

（文責: 柿本翔大）

ログイン画面

アプリケーション起動時にログインを選択した場合，ログイン画面へと遷移する．誤ってログイン画面へと遷移した場合，1 つ前の画面に戻ってからアカウント作成画面へと遷移するのは手間で

あると考え，ログイン画面とアカウント作成画面には，直接相互に遷移できるボタンを用意した．ログインする場合ユーザーは登録してあるニックネームとパスワードを入力し，「ログインする」と表示されたボタンを押下することで入力情報をサーバーに送信し，入力情報が正しければそのアカウントの ID をサーバーから受け取り，ログインできる．エラーチェックに関しては，アカウント作成機能と同様に入力文字数や文字種などサーバーに送信しなくてもエラーと判断できるようなものが入力情報に含まれている場合はサーバーに送信せず端末側でエラーチェックを行った．

（文責: 柿本翔大）

ログアウト機能

ログアウトはユーザー設定画面から行うことができる．ログアウトを選択すると本当にログアウトするかユーザーに尋ねるダイアログが表示される．これは，ユーザーが誤ってログアウトすることを防ぐためだ．ログアウト確認ダイアログでログアウトするを選択すると端末に保存されている ID をサーバーに送信し，ログアウト処理を行う．サーバーでログアウト処理が正常に終了したことが確認できたら，端末に保存されている ID やニックネームなどのユーザーデータを削除し，初期画面に遷移することでログアウト完了となる．

（文責: 柿本翔大）

友だち検索画面

友だち検索画面では，入力した文字列を部分文字列として持つユーザーの名前を検索できる．検索欄に文字を入力し，ソフトウェアキーボードの検索ボタンを押下するとキーボードが非表示になり，入力情報とユーザー自身の ID がサーバーに送信され，検索結果が表示される．検索ボタンを押下した際に何も文字が入力されていない状態であった場合，サーバーとの通信は行わずに名前の入力を促すメッセージが表示される．文字が入力されていても検索に該当するユーザーがいなかった場合，ユーザーに結果を知らせるために該当ユーザーが見つからなかった旨を知らせるメッセージが表示される．正常に検索が行われた場合，RecyclerView によってリスト形式でユーザーが表示される．各ユーザーの表示項目は 3 つあり，アイコン，ニックネーム，検索をしたユーザーの状態に対応したボタンである．検索をしたユーザーが相手ユーザーに対し，まだ友だち申請を送っていない状態であった場合は「申請」ボタンが表示され，友だち申請を送っているが，相手ユーザーが承認をしていない状態であった場合は「承認待ち」ボタンが表示される．「申請」ボタンを押下すると友だち申請を送ることができ，「申請」ボタンは「承認待ち」ボタンへと変化する．「承認待ち」ボタンを押下した際の動作はまだ実装できていないが，ダイアログによる確認をユーザーに対して行ったうえで，友だち申請を解除し，「承認待ち」ボタンが「申請」ボタンへと変化するようにする予定だ．「申請」ボタンを押下した際に相手ユーザーから友だち申請が届いていた場合は，友だち申請を承認したとみなし，「申請」ボタンは「すでに友だち」ボタンへと変化する．なお，友だち検索機能はユーザーを検索して友だち申請を送ることを目的とした機能であるため，検索した際すでに友だちになっているユーザーは表示されないようになっている．

（文責: 柿本翔大）

友だち一覧画面

友だち一覧画面では，友だちになったユーザーの一覧と，ユーザー自身に対して友だち申請をしたユーザーの一覧を見ることができる．両者に共通して各ユーザーのアイコン，ニックネームが表示され，それらに加えて前者は位置情報が，後者は送られた友だち申請を承認，もしくは拒否するボタンが表示される．これらはもともと RecyclerView の viewType を用いてレイアウトを切り替えて 1 つの画面に同時に表示していたが，友だち申請が多く送られてきた際，友だちになったユーザーの一覧が見づらくなるのではないかというレビューをいただいた．そのレビューをもとにして，TabLayout と ViewPager2 を用いて画面を左右にスワイプすることで友だちになったユーザーの一覧と，ユーザー自身に対して友だち申請をしたユーザーの一覧を切り替えて表示できるように変更した．この変更によって画面がよりシンプルになりそれぞれの一覧は見やすくなったが，画面をスワイプしなければ送られてきた友だち申請に気付かない可能性が出てきた．そこで，友だち申請が送られていた場合に，友だち申請が届いている旨とその件数を友だちになったユーザーの一覧が表示される画面にメッセージとして表示することで，見やすさと使いやすさを両立させることができた．

（文責: 柿本翔大）

5.3 サーバーサイドアプリケーション

5.3.1 開発環境

開発にあたり，以下のツールおよび言語を用いた．

- Visual Studio Code 1.63.2
- Docker 20.10.11
- Rust 1.55
- MySQL 8.1
- Python 3.10.0

Python や Go での開発経験を持つメンバーはいたが，Rust での開発経験を持つメンバーはいなかった．しかし，Rust のパフォーマンスの高さと，近年の Rust の人気 [45] を考慮し，Rust を採用した．

（文責: 山本竜生）

5.3.2 Docker, Docker Compose

サーバーサイドアプリケーション（以下サーバー）は，サーバー開発者に加えて，モバイルアプリケーション開発者からも起動される．通常，モバイルアプリケーション開発者はサーバー用の環境構築をしていない．環境構築の手間を減らし，簡単にサーバーを起動するために，Docker と Go 製のラッパーツールである Docker Compose を採用した．Docker とは，「開発者やシステム管理者が，コンテナでアプリケーションを 構築（build），実行（run），共有（share）するためのプラッ

トフォーム」[46] である．コンテナとは，Linux の機能とシステムコールを用いた仮想環境である．Docker Compose は，複数のコンテナを定義し実行する Docker アプリケーションのためのツールであり，設定内容にもとづいたアプリケーションサービスを生成し，起動する．

Docker を使用することによって，ほかの開発者は Docker が動作する環境を構築するだけでよい．Docker で開発環境を用意した場合，作業者の開発環境以外でも動くことが期待されるため，本番環境へ簡単に移行できる，個別にトラブルの対応をしなくてよくなるなどの利点がある．5.3.4 節の API サーバーでは，実際にサーバーを稼働するためのコンテナと，サーバーが正しく動いているかテストするコンテナを用意した．これら 2 つのコンテナは，Docker Compose のサービスとして構成されている．これにより，誰でも簡単にテストをできるようになり，透明性の向上につながった．

API サーバーと同じく，データベースサーバーも Docker Compose のサービスとして構成されている．Docker Compose を用いることで，各コンテナをサービスとして登録し，複数のサービスが関係するアプリケーションの実行を制御できる．5.3.4 節でも後述するが，API サーバーとデータベースサーバーは，Docker Compose のネットワーク機能によって 1 つの仮想ネットワーク内に配置されている．これにより，2 つのサーバーが，いつでも同一の設定で通信することが可能になる．これは，Docker および，Docker Compose で開発環境を用意する利点の 1 つである．

（文責: 山本竜生）

5.3.3　モックサーバー

スプリント期間中，API サーバーを実装するまでの間はモバイルアプリケーションで通信が絡む機能を実装できないため，先にモックサーバーを用意した．モックサーバーは状態を持たず，同じ入力に対して，常に同じ出力をするというものだ．モックサーバーが API サーバーより先行することで，モバイルアプリケーション側では，API のエンドポイントと入出力の定義がわかる．大まかな挙動がわかる．などの利点がある．

モックサーバーは Python 製の FastAPI というフレームワークを用いて実装した．FastAPI を選定した理由は，以下の通りである．第一に，通信に関わる JSON の型を Python の型アノテーションという機能によって決めるため型が比較的安全であるから．第二に，同じく軽量な Python 製 Web フレームワークの Flask よりも高速に動作 [47] するから．第三に，SwaggerUI というインターフェイスを実装しており，API の定義がブラウザ上ですぐに確認できるからである．

モックサーバーに機能を実装したら，Google Compute Engine[*1] 上に配置し，モバイルアプリケーション開発者がローカルにサーバーを用意する必要がないようにした．また，SwaggerUI により，API 定義を Web 上で確認できるようにした．これらは，開発の効率化に大きく貢献した．

モックサーバーを運用するにあたり手間がかかる部分は，モックサーバーを実装した後に仕様が変更された場合，モックサーバーと API サーバーの両方を変更する必要があることである．私たちはモックサーバーをドキュメントとして使用していたため，迅速に更新する必要があった．更新漏れや定義の齟齬が発生すると，どちらに合わせて実装するかが不明瞭になるため，正しく開発できない恐れがある．

[*1] Compute Engine: 仮想マシン（VM） | Google Cloud https://cloud.google.com/compute

（文責: 山本竜生）

5.3.4 API サーバーおよび，データベースサーバー

実際にモバイルアプリケーションと通信するのが API サーバーである．API サーバーは Rust と Rust 製のフレームワーク「Axum」を使用して開発した．Axum は，Rust で主に使用される非同期ランタイム「tokio」と同一のチームが開発していることから採用した．

API サーバーでは，ユーザー情報の登録・友だち情報の管理など，状態が関わる処理を行っている．API サーバーが行っている主な処理は以下である．

- ユーザー情報の登録
- ユーザー情報の更新
- 友だち情報の管理
- ユーザー情報返却
- 友だち一覧返却

ユーザー情報には，名前や現在の位置などが含まれている．位置情報は，スマートフォンで検知したビーコンをもとに取得している．検知したビーコンの識別には，LATTE のために用意された学内のビーコンのリストを流用した．LATTE とは，FLAT と同じくビーコンを用いた学内向けサービスで，サーバーに情報を送信することなく，自身の学内での滞在場所と時間を記録できる．ビーコンのリストを FLAT のリポジトリへ取り込む際には，Git Submodule という機能を使用した．Git Submodule とは，あるリポジトリに別のリポジトリを取り込む機能だ．ビーコンのリストは，機密性が高い情報を扱っており，公開してはいけないという制約があった．Git Submodule を使用することで，ビーコンのリストのリポジトリを閲覧する権限がない人は，FLAT のリポジトリからビーコンのリストを閲覧できないしくみを実現した．

友だち情報には，あるユーザーのすべての友だちが含まれており，各友だちについて，ユーザー情報のうちから必要なものを取り出して送信している．

API サーバーに実装された各機能はユニットテストを行っている．テストを行うことで，事前にバグを発見できる．実装漏れを発見できるなどの利点がある．実装が難しい部分は，先にテストを書き，テストに適合するコードを書いた．コードよりも先にテストを書く開発手法をテスト駆動開発（TDD）と呼ぶ．理想の動作を把握することで，見通しが良くなり，正しいコードをすばやく実装する目的がある．今回，テストを書くにあたり，『はじめて学ぶソフトウェアのテスト技法』[48] を参考にした．

API サーバーが処理したデータを保存するために，データベースサーバーがある．使用した RDBMS は MySQL 8.1 である．API サーバーとデータベースサーバー間の通信には，Rust 製の O/R マッパー「Diesel」を用いた．O/R マッパーを用いた理由は，Rust の文法を用いてクエリーを発行できること．テストが可能になる．などがある．Diesel は Rust 製 O/R マッパーのデファクトスタンダードだったため，採用した．今回，RDBMS として MySQL を用いたが．PostgreSQL のほうが Diesel との相性がよいため，PostgreSQL に移行する予定である．

データベースサーバーでは，ユーザー情報，友だち情報，ビーコンのリストを保存している．

ユーザーテーブルは，機密性が高いパスワードの情報が含まれるため，生のパスワードを直接保存せず，ソルトとハッシュ値を保存した．ソルトとは，パスワードに付加するランダムな文字列の

ことである．ハッシュ値は，ソルトとパスワードを結合したものにハッシュ関数を適用したものを用いた．万が一，データベースが流出しても，生のパスワードを直接見ることはできないため，不正ログインをされにくくする効果がある．

友だちテーブルには，acctive と passive というカラムがあり，これらが複合キーになっている．ユーザー A がユーザー B に友だち申請をした場合，acctive にはユーザー A のユーザー ID を，passive にはユーザー B のユーザー ID を格納する．承認された場合は，反対に，acctive にユーザー B のユーザー ID を，passive にユーザー A のユーザー ID を格納する．これらのレコードの状態から，あるユーザーから見た，友だち・承認待ち・未承認の 3 状態を決定できる．

ビーコンテーブルでは，各ビーコンを major と minor という整数値で管理している．これは，前述した別リポジトリのファイルをもとにしている．

（文責: 山本竜生）

5.4 デザイン

5.4.1 ロゴ

ロゴ作成では，デザイン案を大巻・山本が行い，清書を山本が行った．デザイン案では Figma を使用し，清書では Adobe Illustrator を使用した．始めは大巻がデザイン案を考え，その案に対しほかのチームメンバーがレビューをする形式をとった．1 回目のレビューの際に出た意見として，シンプルな文字のロゴか FLAT らしさを感じられるイラストのロゴが良いという意見が出たためレビュー後はこの 2 つの意見に沿ったデザイン案を大巻と山本で思い付く限り考案した（図 5.2）．ロゴを作成する際には FLAT のテーマカラーである鮮やかな緑色（18D7C3）を使用し，シンプルで見やすいロゴを作成した．イラストを用いたロゴでは，本サービスが価値としている「気軽に友だちを訪ねる」場面をイラストで表そうとしていたが，アプリケーションロゴのサイズ上厳しいことが判明したため断念することとなった．2 回目のレビューで考案したロゴ案の中から絞り込みを行い，その後は清書していくこととした．絞り込みの方法として，始めにチーム内で投票を行い 2 つまで絞り，その 2 つのどちらが良いかをプロジェクト全体で投票した．最終的に，「FLAT」の英単語の意味である「平らな，平坦な」を参考に作成した，未来大の各階層を 1 つまとめるイメージのロゴに決定した（図 5.3）．

（文責: 大巻佑太）

5.4.2 アプリケーションデザイン

本チームにはデザイナーがいないため，モバイルアプリケーションのデザインは，山本が担当した．デザインには Figma というツールを用いた．Web または，アプリケーション上で作業ができ，共有や共同編集が簡単なことから採用した．使用する色は事前にカラーコードを指定し，カラーパレット（図 5.4）にまとめた．カラーパレットには，色を統一し，開発者が色を確認しやすくする目的がある．デザインは OS ごとに用意した（図 5.5，図 5.6）．

初めにすべての画面を大まかに作成し，詳細なデザインについては漸進的に行った．各スプリントのスプリントプランニングで作成する画面が決定したら，その画面についての画面遷移やポップ

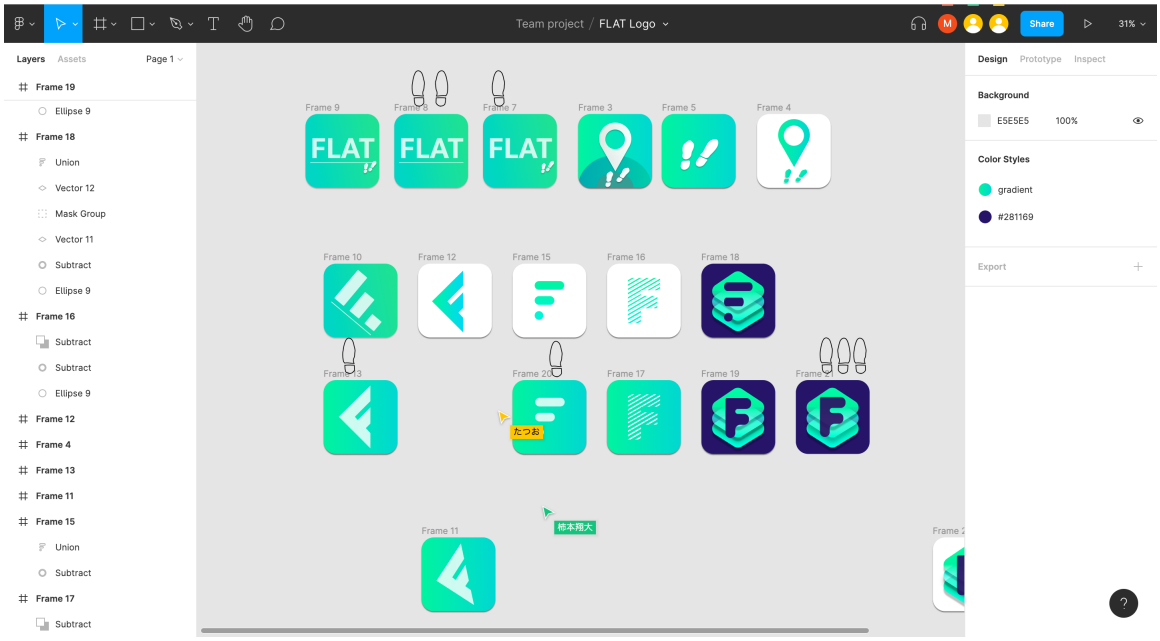

図 5.2　考案したロゴ

図 5.3　決定したロゴ

アップ，各パーツの配置などを詳細に決定した．これは，デザインにかかる時間を分散し，ほかの作業の時間を確保する目的がある．

デザインする際には，ユーザーがストレスを感じないような，情報の配置や画面遷移を心がけた．デザインが完成し次第，ほかのメンバーにレビューをもらった．積極的にフィードバックをデザインに反映させ，ユーザー体験の向上に努めた．

現在は，ユーザーからのレビューを受けることができていない．今後は，実際にユーザーに使用してもらい，フィードバックを反映させ，さらなるユーザー体験の向上に努めたい．

（文責: 山本竜生）

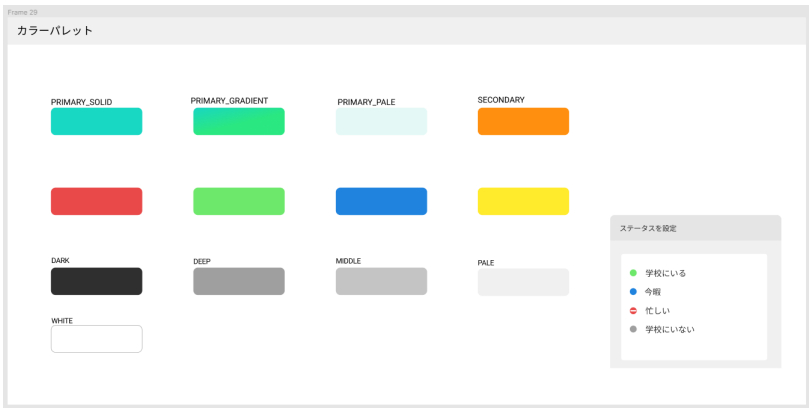

図 5.4　カラーパレット

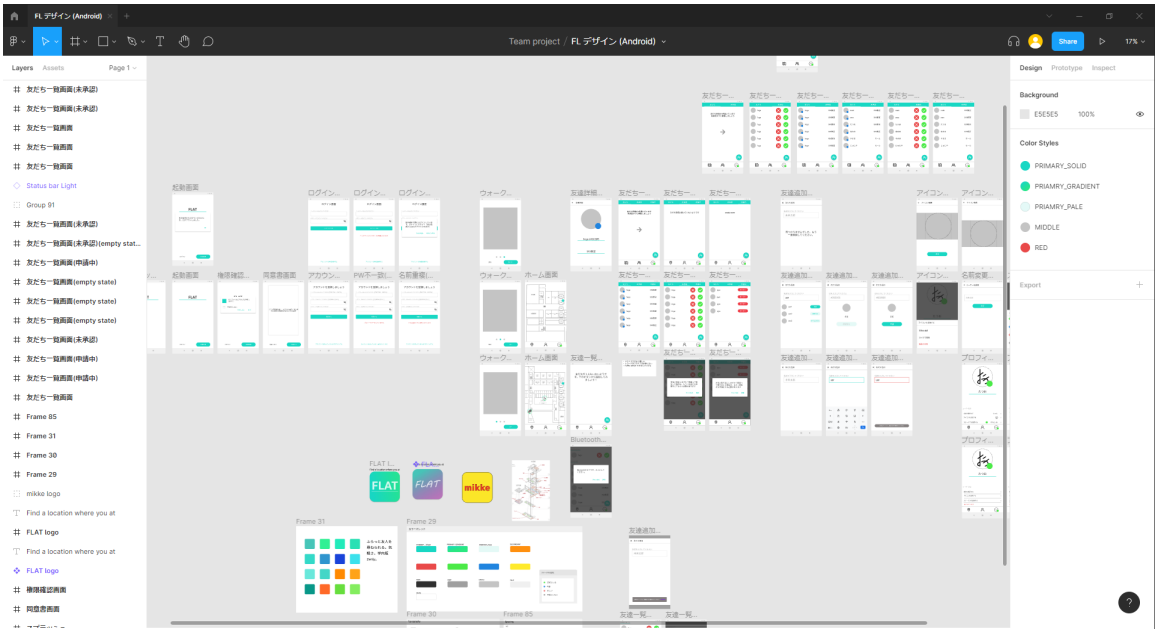

図 5.5　Android 向けデザインの一部

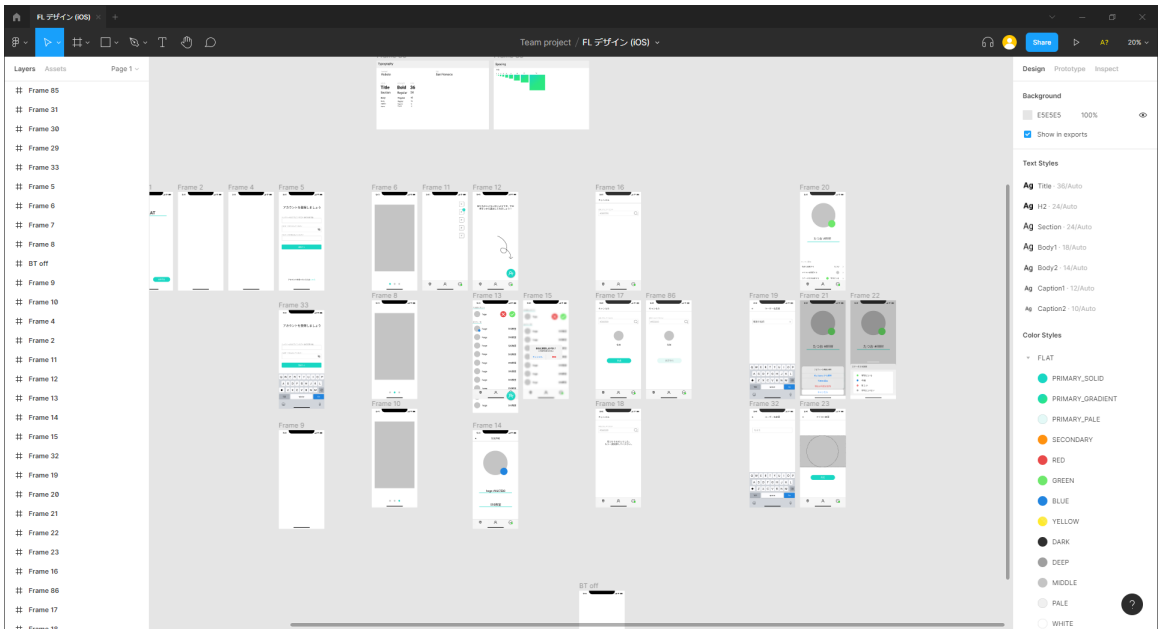

図 5.6　iOS 向けデザインの一部

第 6 章　課外発表会

2021 年 12 月 18 日に enPiT2 BizSysD 北海道・東北合同発表会がオンラインで開催された．この発表会は，会津大学，岩手県立大学，公立はこだて未来大学，室蘭工業大学で実施された各 PBL の合同発表会である．本プロジェクトからは，本サービス FLAT を開発したチームのみが参加した．

合同発表会への参加にあたり，ショートプレゼンテーション用のスライド（図 6.1）と，サービス紹介用のポスターを作成した．ショートプレゼンテーションは 1 分以内で発表のタイトルと概要を説明するものであったため，スライドにはサービスの価値・体験を短くまとめた．ポスターに関しては 2021 年 12 月 10 日に開催されたプロジェクト学習成果発表会の際に作成したポスターをそのまま使用する想定であると告知されたが，プロジェクト学習成果発表会の際に作成したポスターは本プロジェクト全体に関するポスターであり，参加チームが 1 チームであることと，サービスに関する詳細が書かれていないことなどの理由からサービス紹介用のポスターを新たに作成した．サービス紹介用のポスターには，そのポスターのみを用いて発表が行えるようにサービスの概要や機能などサービスに関する内容のほかに，開発手法やチーム構成，ビーコンについての説明，プロジェクト学習を通して得たメンバーの学びなどをまとめて記載した．

合同発表会当日は全参加チームのショートプレゼンテーションが行われた後，前半 10 チーム，後半 9 チームに分かれて各 50 分の発表が行われた．後半チームの発表終了後に投票時間が設けられ，発表者，聴講者を含めた参加者全員での発表に対する投票が行われた．以下は投票で使用された評価フォームにていただいた応援コメントである．

- やっていること自体面白く感じたから。
- 是非実用化して欲しいです！
- レベルの高い発表を聞けて良かったです！
- 先生の位置情報がわかるといいなあと思う瞬間があるので，応援しています．
- ビーコンの復習できました。

投票時間の後に開催された閉会式では，投票の集計結果が発表された．本チームは，PBL を通して最も多く学びを得たと思われるチームに与えられる「学び賞」を得た．合同発表会に参加した経験やいただいたコメントを励みとして，2022 年 2 月 15 日に開催される課外発表会へ向けてサービスの実装や発表準備などを進めていきたい．

（文責: 柿本翔大）

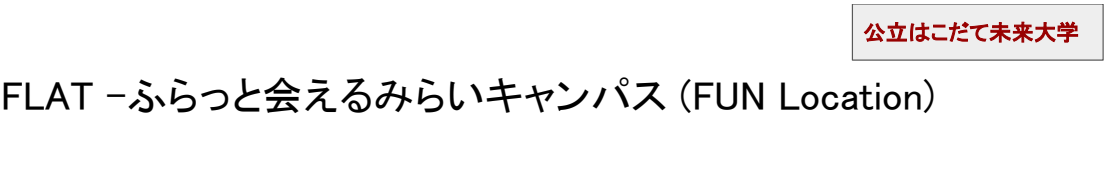

図 6.1　ショートプレゼンテーション用のスライド

第 7 章　各メンバーの学び

7.1　大巻佑太

プロジェクト活動を通して，さまざまな経験をできた．前期の活動の中盤ごろからファシリテーターを専任していた．ファシリテーターに立候補した理由として，週替わりでファシリテーターを担当すると進行が遅くなってしまうと感じたことや，自分の長所だと感じているコミュニケーション能力を活かせると考えたからだ．しかし，1 人でファシリテーターを務めていく自信がなかったため補佐 2 名をメンバーから指名した．補佐を決める際に重要視したのは，「自分に対してはっきりと意見を出せること」と「自分がミスをしたときに指摘してくれること」の 2 つである．当時はまだプロジェクト活動が始まってあまり時間が経っていなかったが，短い期間の中で多くの発言をしていたメンバーを指名することにより，ファシリテーターとして活動しやすい基盤を作ることができたと感じている．

しかし，ファシリテーターとして活動していくうちに大きな問題点が発覚した．「アジェンダを決めなければならない」というものだ．これにより，予期せぬ仕事ができてしまったため責任が重くなってしまった．これは，本プロジェクトチームにプロジェクトリーダーが存在しないがために起こってしまったことだと私は考えている．説明責任を分担するために大臣制度を取り入れて活動をしていたが，前期の時点でチームに浸透しておらず仕事のぶん担ができていなかった．結果として，ファシリテーターがアジェンダに対して責任を持つことになってしまったと考えている．問題を解決するため，ファシリ会議にほかの人も参加してくれるよう呼び掛けたりしたが，なかなかうまくいかず歯がゆい想いをした．結局，前期の活動で改善できず，後期からは各チームごとの作業が多かったため最後までアジェンダ決めをすることとなってしまった．今考えると，「アジェンダ大臣」などを選定するために，自分からメンバーに呼びかけるべきだった．

ファシリテーターを専任してからは，多くの課題が見つかった．実際に進行していて感じたのはレスポンスの無さである．12 人メンバーがいる中で発言が多いのは 3・4 人程度で，それ以外のメンバーからの反応がなかったり，本当にやるべきことを理解できているかを判断するのが難しく，苦心した．対面での活動であれば，相手の顔を見て判断できるし，メンバー側としても気軽に発言できるため進行しやすい．しかし，活動の大部分がオンラインだったため，カメラ OFF・ミュート状態の人も多い．そのため，どうすれば発言を促すことができるかが課題だった．さらに，意見交換をする方法がオンラインだとブレイクアウトルームに分かれないと発言が被ってしまったり，そもそも発言数が少なくなってしまうため，どんな活動でもブレイクアウトルームに分かれなければいけないのも大きな課題だった．課題を解決するため，進行中，都度メンバーに理解できているか確認をとったり，事前にブレイクアウトルームに分かれるタイミングや，作業時間・作業内容を決めておいたりする方法をとった．これにより，何をすべきかわからないメンバーが減り，ブレイクアウトルーム内での意見交換が行いやすくなったと感じた．しかし，全体でのレスポンスの少なさはあまり改善できなかった．初めと比較すると増えはしたが，それでも半数程度という結果になってしまったのは反省点である．

サービスのアイデア出しが始まってからは，アジェンダの決定が難しく，うまく進行できないこ

とが多かった．アイデア出しに関するアジェンダを全員の意見をもとに決定すればよいのか，ある程度自分で決めてよいのかがわからず時間を無駄にしてしまったため，アイデアの決定を中間発表までにできなかった．これは，アイデア出しの最終期限の見積もりができていなかったことや，絞り込みの方法をあらかじめ決定していなかったことが原因だと考えられる．アジェンダを決定していたこともあって，責任を感じる結果となってしまった．

後期の活動が始まってからは，各サービスごとのチームでの活動が主となったので，ファシリテーターとして活動する機会は少なかった．しかし，期末発表・提出物に向けた大臣の選定や，スケジュールの確認などをもっと確認しておくべきだったと感じている．中間発表の段階で，期日直前に作業していたため，早めに行動したほうが良いと感じたはずなのにもかかわらず，大臣が活動するまで受け身の姿勢でいたことは大きな反省点である．

ファシリテーターを務めたのは私にとって大きな経験になった．自信の長所だと感じるコミュニケーション能力を大きく伸ばすことができたし，多くの課題を発見し，解決策を考える機会が非常に多かったため課題解決力を身につけることができたと感じている．責任も大きかったため，ファシリテーターをやり切ったことは誇りに感じている．

サービス開発では，プロダクトオーナーを担当した．理由としては，チームメンバーでロールについて話し合った際に，私に適性があるからと勧められたからである．私自身，アプリケーション開発の一連の流れを深く理解したいと思い，ビーコンプロジェクトに所属したため，全体を見ることができるプロダクトオーナーを引き受けた．プロダクトオーナーが何をするのか全然わからないところからのスタートで不安も多かったが，長期休みを利用してプロダクトオーナーについて勉強をした．勉強方法は，「さまざまな記事を読む」「別チームのプロダクトオーナーと話し合う」「プロダクトオーナー経験者から助言をいただく」などである．勉強を進めていき，プロダクトオーナーについての理解を深めていくのは楽しかった．

勉強したことによって，後期プロジェクト開始からすぐにスプリントに入ることができた．プロダクトバックログの優先順位を考えるのが難しかったが，一人で悩むのではなく意見をもらう判断ができて本当によかった．初めはバックログの粒度があまり良くなく，スプリントゴールを達成できなかったため，バックログを減らすことができず不安に感じていた．しかし，スプリントレトロスペクティブを行うことで全員が意見を出し合い，次のスプリントから改善していくことができたため，頼もしく感じた．基本的にプロダクト全体を見ることに専念していたが，プレゼンの資料を作成したり，サービスロゴのデザインをした．プロダクトの価値がしっかり伝わるような資料の作成は難しく，初めはうまくできなかったが，回数を重ねるごとに伝え方が良くなるのを自分でも理解できて，うれしく感じた．

プロダクトオーナーを務めたことによって，プロダクトの価値について考える機会が非常に多く，「どうすればユーザーにとって価値があるか」「このサービスは何を売りにしたいのか」などさまざまな視点で物事を考えることが身についた．それにより，レビューで意見を出しやすくなったのがとてもうれしい．

プロダクトオーナーを務めていて感じた反省点はいくつかある．初めに，プロダクトオーナーはデイリースクラムに参加することは必須でないことから参加しなかったことだ．その結果，自分が行っている作業の透明性が失われてしまった．次に，今何をしているか，何が終わっているかをチームメンバーに共有することが疎かになってしまったのは大きな反省点である．実際に，チームメンバーから指摘を受け，改善できたが，指摘されるまで気付かなかったのは本当に良くないと感じている．そのほかには，動画撮影の際にユーザーストーリーを盛り込むことができなかったこと

や，期末発表の発表時に初めが早口になってしまったことである．

チーム開発の経験がなかったため，コミュニケーションの大切さやふりかえりの大切さなど，さまざまなことを学ぶことができた．発表会では，私がプロダクトオーナーを担当していたため，スライド作りや原稿などの作成した．これにより伝わりやすい話し方や内容，スライドの配置など実用的なことを多く学ぶことができた．いただいた評価の中に，「実際に使ってみたい」とあったことが本当にうれしかった．中間，期末ともに動画・ポスター作成が難航したことはプロジェクト全体で反省すべき点であると考える．メンバー間での情報共有や，仕事のぶん担ができていなかったため大臣制度がうまくいっていなかったと感じた．

私が担当したサービスは，私の出した意見がもとになっているためプロダクトオーナーとして，サービスの価値と向き合い続けることができたのはとてもよい経験になった．チームメンバーとどのようにすればサービスがもっと良くなるか，円滑に開発を進められるかを話し合い，実際にサービス開発に落とし込んでいくことができた．さらには，先生や TA の方々からレビューをもらい，その内容をもとに議論をするなど，今までにない経験ができた．実際にサービスをリリースするまで，今のメンバーとともに開発を続けることになっているため，納得できるまで活動していきたい．さらに，未来大生に使用してもらい，多くの感想を聞いてみたいと感じている．

プロジェクト学習では，本当に多くのことを学ぶことができた．自身の長所を伸ばすことができたし，まだ身についていないものも浮き彫りになったため，今後改善していくことができる．配属当初は，自身が責任の大きな役職に就くことはまったく考えていなかったが，実際は進行を任される立場になったりと，とても有意義な活動になったと感じている．一番大きいのは，やり切ったことにより自信がついたことである．わからないことが多くある中皆でやり切ったことが本当にうれしい．この経験を，今後の活動にも活かしていきたい．

（文責: 大巻佑太）

7.2　小田嶋亜美

私はこのプロジェクト活動で FUN Location チームの開発メンバー，成果発表会のポスター責任者の役割を受け持った．このプロジェクトに入ったきっかけは，今までやったことのないアプリケーション開発に挑戦してみたい，実際使えるアプリケーションを開発してみたいと思ったからだ．興味本位で入ってしまったため，アプリケーションは何をすればできるのか，使う言語は何なのかもまったく分からないままこのプロジェクトに希望した．最初のプロジェクトの説明や過去のプロジェクトの活動記録などを見て，アプリケーション開発をするためにチームを作らなければならない，また，チーム開発をするためにさまざまな開発手法があることを知った．短い期間の中で多くの作業をしなければならないことに最初は絶望したが，これも自分を成長させるためだと思い気合いを入れた．活動の中で，アイデアを絞るのにこんなに時間がかかるとは思わなかった．出したアイデアに基準をつけて，何回も検討し合う中で，1 つのサービスを考える過程がこんなにもたいへんなことだとは思わなかった．チームに別れ，私は開発メンバーの役割を任された．もともとプログラミングが苦手な私にとって，今までやったことのない言語を使って開発することに不安を感じていた．勉強するにも最低限必要な知識は何なのか，開発環境をどうやって作ればよいのか，すべてが分からないことだらけだった．開発期間に入っても，分からない部分を調べるのに多くの時間を費やして，作業がなかなか計画的に進まなかったり，開発で使った Git，GitHub の使

い方も同じチームメンバーに教えてもらったが，コマンドを打ち間違えたり，コードを共有しているため，自分が Git の操作を間違え，ほかのメンバーに迷惑をかけてしまった．私は iOS アプリケーションを担当し，Swift 言語を初めて使った．普段の授業で教わった C 言語や Java しか知らなかったためどのようにコードを書けばよいのか分からなかった．さらに，オブジェクト指向などの開発に必要なコードの書き方なども最初は分からなかったので．チームメンバーに注意されながらも開発した．いつもプログラミングが苦手だという理由でコードを書く作業から逃げてきた私にとって最初の 2，3 週間はとてもつらかった．何をやってもうまくいかない，どんどん周りの人達との差がついていく，そんな状況に何度も挫けそうになった．チームメンバーもプロジェクト活動で初めて知り合った人達だったので，最初はコミュニケーションがうまく取れず，遠慮してなかなか自分から話せなかったり，ほかのメンバーが話していることを理解できなかったりした．しかし，開発をチームメンバーで協力して行うことをこのプロジェクト活動で学ぶことができた．悩みや問題をほかのメンバーに相談し，一緒に解決策を考えてくれた．最初は問題点を話すとほかのメンバーに迷惑をかけるのではないか，作業時間を削ってしまうのではないかという理由でなかなか自分から話すことができなかったが，チームメンバーが親身になって私の問題点に時間割いてくれ，一緒に考えてくれた時はとてもうれしかったし，チーム開発の良さ，楽しさを感じることができた．初めてアプリケーション開発を経験したけれども，私のような何も分からない初心者に対しても開発を進めていく中で一番チームメンバー間のコミュニケーションが一番重要だと気付いた．

初めてアプリケーション開発を経験してうまくいかなかったことはたくさんあった．第一に，新しい言語の習得だ．今まで自らプログラミング言語の勉強をしてこなかった私にとって勉強することはとてもたいへんだった．参考書を読んでも出てくるプログラミング特有の語句の意味が分からなった．その度に何度も調べたりして，実際にコードを書いて作業する時間がなかなか取れなかった．第二に，プログラミング界隈の用語だ．チームメンバーが話している中で私にとってよく分からないことについて話していることがあった．開発作業に必要なことだとは分かるが，それは何を意味しているのか，特にサーバーサイドについてまったく分からなったので API を使う作業の時はとても苦労した．第三に，チームメンバー間のコミュニケーションの難しさだ．プロジェクト内で女子が私一人しかいなく，気軽に話せる人がいなかった．人と話すのは好きだが，分からないことがあった際，相手に迷惑をかけるのではないかと思ってしまい自分から発言があまりできなかった．また，プロジェクト活動内で大量の開発に関する情報が入ってくるため，すべてを理解し，吸収することが難しかった．同じことを何度も聞いてしまったり，情報が抜けていて開発作業に遅れをとってしまった．最後に，タスクの見積もりの甘さだ．タスクに重み付けをし，最優先のタスクから順番を決めていたが，進捗がなかなか生まれず，軽いタスクから手をつけてしまったり，いろいろなタスクに中途半端に手をつけてしまったりした．

チームでの開発を通して成長できたことは 2 つある．第一に，チームメンバーとのコミュニケーションだ．最初は自分から発言ができず，他のメンバーのしっかりと理解しているのかを見えなかった．開発期間に入ると，私にチームメンバーがもっと発言してほしいと指摘を受け，開発の中で問題や悩んでいることを Slack で質問したり，Discord で通話をつなぎながら作業したりしてチームメンバーに遠慮なく話かけることができた．また，レトロスペクティブで決めた Try によって夜会でのモブプログラミングで開発作業を手伝ってもらったり，朝会後に雑談を交えながら，問題点を一緒に解決してくれた．それによって，なかなか自分から発言できなかった雰囲気を改善し，開発を進めていく中で自分から話し，相談することの重要さを学ぶことができた．第二に，自発的に学ぶ姿勢だ．ほかのチームメンバーに比べて，開発経験がまったくなかったので，分からな

いことは自発的に調べるようにした．朝会でやることを明確にし，作業しながらもほかのチームメンバーが Slack で話している内容などをメモして自分の作業に役立てたり，タスクの参考になりような Web ページを Slack に投稿して，再度確認し，作業に役立てた．開発が進んでいくと，自分から人を集めて作業を手伝ってもらったり，先輩方に質問したりして，学ぶ意欲を高めた．Slack の自分のチャンネルに bot を作り，作業に圧をかけながら，何とか期間内にタスクをすべて終わらせようと努力した．ほかのチームメンバーから一番成長している姿が見えると言われた時は，自分の努力がほかの人にも見えている事を知り，とてもうれしかった．

改善点は，もっと開発に関する用語を学ぶこと，作業の効率化をすることである．私は，今でもプログラミングに対して苦手意識を持っているため，プログラミングが楽しいと思えるようにもっと言語の勉強やサーバーサイドの用語の理解をしていきたい．私もチームメンバーと同じように開発，プログラミングに関する話題の会話に入れるように自分自身を，もっとレベルアップしていきたい．作業の効率化では，タスクの重み付け，優先タスクの可視化をしっかりと行い，作業の見積もりしていきたい．まだどのタスクにどのくらいの時間がかかるのかがあまりつかめていない状態なので開発作業に慣れ，期間内にすべてのタスクをこなせるように時間の使い方などを意識して効率化を行っていきたい．

（文責: 小田嶋亜美）

7.3 柿本翔大

1 年間のプロジェクト学習からは多くのことを学んだ．まず，前期には主にサービスの考案をした．考案プロセスとしてはフィールドワークを行い，その結果をもとにしてオープンスペーステクノロジーによる議論をし，評価基準を決定することでアイデアを絞り込むというシンプルなものだった．これまでハッカソンなどの短期間で運用を考慮しない少人数によるアプリケーションの開発にしか携わったことがなく，手法としてはブレインストーミングを行ったことがあるという程度であったため，最初はなすことすべてが新鮮に感じた．しかし，サービスアイデアはなかなか決定せず，次第にプロジェクト学習の時間が憂鬱に感じることが多くなっていった．私は，自分が使いたいと感じるサービスを開発したいと考えており，出されるサービスアイデアのうち，そのようなものはだいたい過去のビーコンプロジェクトで開発されていたのだ．過去に開発されたサービスの改修を視野に入れず，新規サービスの開発のみを考えていたため良いサービスアイデアが浮かばず，私のモチベーションはとても低い時期が続いたが，それでも活動には積極的に参加した．結果として，自分が使いたいと感じるサービスの開発に携われたので，私の努力は無駄でなかったように感じる．

開発では，スクラムマスターと開発メンバーを兼任した．開発手法としてスクラムを用いると決定した段階では，プロジェクト学習の開始と同時期に参加した別の PBL でもスクラムを導入していたため，ほかのプロジェクトメンバーに比べると少しだけスクラムに関する知識を多く有していた．しかし，スクラムの大半のイベントは経験したことがなかったため，あいまいな部分が多かった．そのため，スクラムマスターとしての責務を全うするには，知識を補う必要があると考え，夏季休業中にスクラムについて勉強した．勉強に用いた資料のうち，わかりやすくまとめられていたり，重要だと思ったものはすべてチームメンバーに共有し，チームメンバー全員のスクラムに対する理解度の向上を図った．スプリント期間中の各種イベントを初めて実施する際には，あらかじめ

そのイベントを行う目的や具体的な進行方法などをまとめた資料を作成し，共有しておくことで円滑な進行を試みた．また，ほかのチームも参考にしやすいように，私たちのチームで行った活動に対してはなぜその活動をしたのかを記録として残した．その結果，私たちのチームでは活動に対して目的意識をもって臨むことができ，有意義に時間を使うことができた．しかし，何のためにその活動をしているのかを考えることすらせずに私たちの活動の形だけをなぞるチームが出てきてしまったのは残念だった．

開発メンバーとしては，Android 開発を 1 人で担当した．Android 開発をしている人がビーコンプロジェクト全体で私 1 人であったため，相談や知識を共有できる相手はいなかった．それに加えて，私自身 Android 開発に関して初心者であったため，そのままでは開発などできないと考え，夏季休業中に Android 開発について勉強し，資料を作成した．実際の開発では，新たに調べなければならないことばかりで多大な時間を費やしたが，開発への導入は円滑に行うことができた．これは夏季休業中に開発の基礎やアーキテクチャに関して勉強した成果である．しかし，スクラムマスターの兼任によって開発に集中できないこともあり，開発がうまく進まないときはとても苦しかった．休む時間をうまく捻出できず，精神的に病んでしまうことがあったため，スクラムマスターと開発メンバーの兼任はどちらかに精通している場合を除き，すべきでないと考えている．

開発全体を通して，作業の透明性やチームビルディングの大切さを実感した．開発が短期間であったためにたとえ 1 日でも問題の発覚が遅れるとそのスプリントではスプリントゴールを達成できないといったことが起き，チーム全体にとって大きな損失となる．そのため，着手中の作業を可視化することや問題が発生した際にすぐに報告することで透明性の確保をすることが重要であると感じた．また，雑談しやすい環境が形成されていることはメンバーのモチベーションに直結し，リラックスした状態で話し合いに臨めるため，さまざまな意見が出てきやすかった．チームメンバー間で思ったことを言い合える仲になると，作業を休む旨の連絡もしっかりと行われるようになり，前述した作業の透明性にもつながった．以上のことから，チームビルディングがとても大切だと感じた．

プロジェクト学習全体を通して，技術的なことだけでなく，チーム開発で大切なことや，人間関係に関することなどさまざまなことを学んだ．ここで学んだことは卒業研究や仕事などで自分の今後に活かしていきたい．

（文責: 柿本翔大）

7.4　山本竜生

普段の活動では，積極的に発言をする人が，私を含めた数名に固定されてしまい，グループ全員で集まる必要があるのかと疑問に感じたことが何度もあった．オフラインでは，直接顔を合わせていたため，相手の反応を簡単に見ることができたが，オンラインになり，顔も見えず，声を聞くこともできない状況だった．前期の半ばに，各メンバーの反応がわからない問題に対応するため，原則カメラをつけて講義に参加することを合意した．合意した当初はほとんどの人がビデオをつけていたが，夏休みが明け，後期に入るとビデオをつける人は次第に減っていた．チャット機能を用いたコメントや，リアクション機能を用いることをグループ内で合意し，決定したが，これも用いられることはあまりなかった．全体での活動を通して，チームビルディングの難しさを実感した．本グループは，グループリーダーが存在していないが，このように自主性が低いチームには向いてい

なかったと考える．

前半の活動では，フィールドワークやサービスを考案するために頭を悩ませたのをよく覚えている．私は，20 年間函館周辺に住んでいるが，実際に函館のまちを歩くと，まったく知らない函館をみることができ，とても楽しかったのを覚えている．愛着を持てるサービスを作りたいという気持ちから，サービスのもとになるアイデアがなかなか決まらず，もどかしい思いをした．

後半のチーム開発では，テックリードのようなことを行った．第 0 スプリントでは，モックサーバーやデザインの概形を作成した．通常のスプリント期間中は，サーバーサイドアプリケーションの開発，iOS アプリケーションの開発，モバイルアプリケーションのデザインを担当した．iOS アプリケーションの開発や，Adobe Illustrator を用いたロゴの清書などは初めての経験だったが，積極的に挑戦した．サーバーサイドの開発では，どうすればモバイルアプリケーションを開発しやすくなるかを考えながら開発し，モックサーバーの開発や，API のドキュメントの整備，API で送受信する JSON の型周りの工夫につながった．iOS アプリケーションの開発では，ロジックの整理，UI のコンポーネント化，モブプログラミングのナビゲーターを担当した．Swift を書くのは，今回が初めてだったが，ロジックの整理と UI のコンポーネント化では，自身の過去の React での Web 開発が活きる形となった．モブプログラミングでは，自分の考えを他人に伝えることの難しさを実感した．

個人開発ではあまりできない，規模が大きめのサービスを開発し，さらに，テストや設計などを経験できた．テストは，想定よりも多くの時間がかかり，効果的なテストをするのはとても難しい作業だった．API や画面の設計では，当初の予定と大きく変わった部分が存在し，さらに，考慮できておらず，修正や追加した箇所も複数存在した．

開発期間は約 2 ヵ月間と短く，実装が間に合うか，常に焦りが伴う開発だった．開発メンバー 3 人で，Android，iOS アプリケーション・サーバーサイドアプリケーションを実装したが，常に人員不足を感じていた．うち 1 名は，開発経験がなく，私が教えながらの開発となった．MVP は達成したものの，精神的に苦しい開発期間だった．困難は多かったが，開発者としてとても成長できたと考える．

（文責: 山本竜生）

第 8 章　まとめと展望

8.1　まとめ

8.1.1　夏休みのふりかえり

夏休みには，決まったアイデアをもとにサービスを考案した．モバイルアプリケーションを用いた位置情報共有サービスとして，GPS を利用したものが多くリリースされている．既存の GPS を用いたサービスでは，屋内の詳細な位置までは測定できないという背景から未来大内で詳細な位置情報を共有できるサービスがないという課題を見つけることができた．そこで未来大内で詳細な位置情報を共有できるサービスを使ってもらうことを目的とし，未来大内に設置されたビーコンを用いて，位置情報共有サービスを提供するアプリケーションを開発しようと考えた．開発手法，チーム内の役割を決め，投票で決まった「学内サービス」のアイデアを深堀りし，サービスを決定した．サービスの考案ではインセプションデッキを作成し，チームメンバー全員のアイデアに対する共通認識を深めつつ，プロジェクトの方向性を確かめることができた．また，ユーザーストーリーマップも作成し，開発作業の準備に役立てた．

（文責: 小田嶋亜美）

8.1.2　後期のふりかえり

後期では，主に開発作業を行った．前期で考えたサービスを開発するために，開発前に必要な言語の勉強や開発で用いるバージョン管理ツールである Git，コードのホスティングサービスである GitHub の勉強を各自で行った．開発期間中はスプリントと呼ばれる短い開発期間を設け，その中で開発を進めた．スプリント期間の最終日にスプリントレビュー，その後スプリントレトロスペクティブを行い，サービスの改善を図った．スプリントレトロスペクティブでは KPT 法を用いて，スプリント期間内の問題点や次回のスプリントに向けての Try を出すことによってより良い開発ができるようにした．平日にデイリースクラムを同期型で実施し，その日に行う作業や，スプリントゴールを達成するための障害となる問題の確認をチーム内で行った．また，Try で出た案を次回のスプリントで実際に行い，開発作業の改善に役立てた．開発期間の中で，問題点が出た場合にすぐ Slack や Discord でコミュニケーションをとり，チーム内で協力して問題を解決できた．

（文責: 小田嶋亜美）

8.2　今後の展望

今後の展望として，2 月の外部発表に向けてアプリケーションの実装を考えている．具体的には，学内マップ上で友だちの居場所を表示する機能やステータス機能の実装，ビーコンの検知精度向上を考えている．

（文責: 小田嶋亜美）

参考文献

[24] *What is Slack? | Slack.* https://slack.com/intl/ja-jp/help/articles/115004071768-What-is-Slack. (Accessed on 01/19/2022).

[25] *Git.* https://git-scm.com/. (Accessed on 01/19/2022).

[26] *GitHub | GitHub is a development platform inspired by the way you work. From open source to business, you can host and review code, manage projects, and build software alongside millions of other developers.* https://github.co.jp/. (Accessed on 01/19/2022).

[27] *esa* - 自律的なチームのための情報共有サービス. https://esa.io/. (Accessed on 01/19/2022).

[28] *Miro(*オンラインホワイトボード*) | aslead |* 野村総合研究所 *(NRI).* https://aslead.nri.co.jp/whitepaper/lp/miro.html. (Accessed on 01/19/2022).

[29] *Discord |* 会話や交流が楽しめるプラットフォーム. https://discord.com/. (Accessed on 01/19/2022).

[30] *Figma: the collaborative interface design tool.* https://www.figma.com/. (Accessed on 01/19/2022).

[31] *Zoom Meetings | Zoom.* https://explore.zoom.us/ja/products/meetings/. (Accessed on 01/19/2022).

[32] *Android Studio* の概要 | *Android* デベロッパー | *Android Developers.* https://developer.android.com/studio/intro?hl=ja. (Accessed on 01/19/2022).

[33] *Visual Studio Code –* コード エディター *| Microsoft Azure.* https://azure.microsoft.com/ja-jp/products/visual-studio-code/. (Accessed on 01/19/2022).

[34] *Xcode 13* の概要 - *Apple Developer.* https://developer.apple.com/jp/xcode/. (Accessed on 01/19/2022).

[35] *Illustrator* とは？ アイコンから大判ポスターまで自由にデザイン. https://www.adobe.com/jp/products/illustrator/beginner.html. (Accessed on 01/19/2022).

[36] *Google* スライド - オンラインでプレゼンテーションを作成、編集できる無料サービス. https://www.google.com/intl/ja_jp/slides/about/. (Accessed on 01/19/2022).

[37] *Google* ドキュメント - オンラインでドキュメントを作成、編集できる無料サービス. https://www.google.com/intl/ja_jp/docs/about/. (Accessed on 01/19/2022).

[38] *A successful Git branching model » nvie.com.* https://nvie.com/posts/a-successful-git-branching-model/. (Accessed on 01/04/2022).

[39] *GitHub - petervanderdoes/gitflow-avh: AVH Edition of the git extensions to provide high-level repository operations for Vincent Driessen's branching model.* https://github.com/petervanderdoes/gitflow-avh. (Accessed on 01/04/2022).

[40] *The default branch for newly-created repositories is now main | GitHub Changelog*. https://github.blog/changelog/2020-10-01-the-default-branch-for-newly-created-repositories-is-now-main/. (Accessed on 01/04/2022).

[41] *Managing a branch protection rule - GitHub Docs*. URL: https://docs.github.com/ja/repositories/configuring-branches-and-merges-in-your-repository/defining-the-mergeability-of-pull-requests/managing-a-branch-protection-rule (visited on 01/05/2022).

[42] *Swift - Apple* **（日本）**. https://www.apple.com/jp/swift/. (Accessed on 01/19/2022).

[43] **配信ダッシュボード** / *Android* **デベロッパー** / *Android Developers*. https://developer.android.com/about/dashboards/index.html?platform. (Accessed on 01/18/2022).

[44] *Android Beacon Library*. https://altbeacon.github.io/android-beacon-library/. (Accessed on 01/05/2022).

[45] *Stack Overflow Developer Survey 2021*. URL: https://insights.stackoverflow.com/survey/2021/#technology-most-loved-dreaded-and-wanted (visited on 01/17/2022).

[46] *Part 1* **：概要説明とセットアップ** — *Docker-docs-ja 20.10* **ドキュメント**. URL: https://docs.docker.jp/get-started/index.html (visited on 01/05/2022).

[47] *Round 20 results - TechEmpower Framework Benchmarks*. URL: https://www.techempower.com/benchmarks/#section=data-r20&hw=ph&test=json&l=zijzen-sf&f=1ekg-jz6rk-4fu138-1kw-5jrvsw-hrbf28-29gxz6-hsjew2-4kuqyw-5jw3ka-6bk-0 (visited on 01/05/2022).

[48] リー・コープランド. **はじめて学ぶソフトウェアのテスト技法**. 宗雅彦 訳. 日経 BP, 2005.

付録 A 中間発表会で使用したプロジェクト概要のポスター

図 1　中間発表会で使用したポスター

付録 B 成果発表会で使用したプロジェクト概要のポスター

図２　成果発表会で使用したポスター

付録 C enPiT2 BizSysD 北海道・東北合同発表会で使用したサービス紹介用のポスター

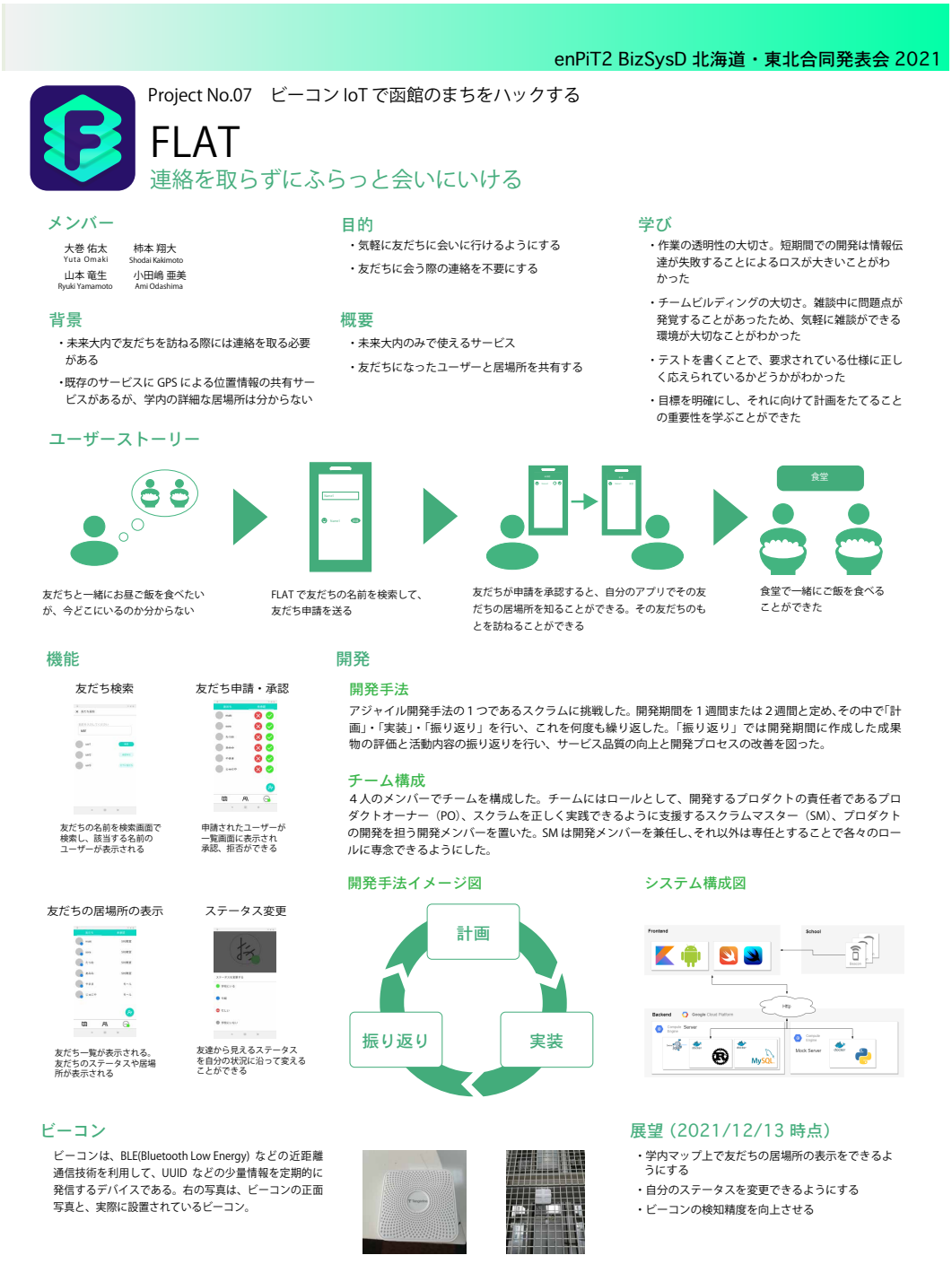

図 3　サービス紹介用のポスター