

愛知工業大学情報科学部情報科学科
コンピュータシステム専攻（メディア情報専攻）

令和5年度 卒業論文

独立したコミュニティにおける
滞在ウォッチの安定運用のための
システム拡張に関する研究

2023年2月

研究者 K19074 外山瑠起
K19036 亀田優作

指導教員 梶 克彦

目 次

第 1 章 はじめに	3
1.1 背景	3
1.2 滞在ウォッチの安定運用のためのシステム拡張	4
1.3 論文構成	4
第 2 章 関連研究	6
2.1 屋内位置推定における在室者の検出方法	6
2.2 在室者の検出方法に関する研究	6
2.3 在室者状況の提示方法に関する研究	7
2.4 コミュニケーション促進に関する研究	7
第 3 章 在室管理プラットフォーム	
「滞在ウォッチ」	8
3.1 「滞在ウォッチ」のシステム構成	8
3.2 在室者情報の可視化	12
3.3 部屋利用者の入退室時刻の評価実験	14
3.4 在室者情報を用いた応用システムについて	16
第 4 章 独立したコミュニティにおける滞在ウォッチの安定運用のためのシステム拡張	18
4.1 安定運用に向けた既存システムの再構築	18
4.1.1 サーバの再設計と再構築	18
4.1.2 クライアントサイドの実装	19
4.2 利用者の管理とアクセス制御システムの整備	19
4.3 スマホアプリによるビーコンと実デバイス	23
4.3.1 BLE ビーコンのアプリケーション化	24
4.3.2 スマホビーコンと実デバイスによるビーコンのハイブリッド活用	25
第 5 章 おわりに	26
5.1 まとめ	26
5.2 今後の課題	26
謝辞の例	28

第1章 はじめに

本章では、研究背景や我々の先行研究について述べ、本研究の目的と概要を説明する。

1.1 背景

近年、社会的に重要な課題の一つに、在室者管理の課題がある。在室管理は、環境保護やエネルギー効率化、セキュリティ強化などの目的から、家庭やオフィスなどの建物内での生活やビジネスに関する様々なアプリケーションに応用される。例えば、エネルギー管理は、居住者がいる場合といない場合でエネルギー消費量が大きく異なるため、在室者管理を使用して、建物内のエネルギー消費を最適化できる。また、セキュリティや照明などのシステムの自動化にも応用され、これにより、環境保護やエネルギー効率化を図れる。在室者管理は、災害時や緊急事態においても重要な役割を持つ。災害時には、避難した居住者を確認が重要であり、在室者管理を使用すれば、確認作業をスムーズに行える。

また在室者管理は、コミュニティにおいても様々なメリットがある。その一つに、コミュニティ内でのコミュニケーション促進が挙げられる。在室者管理を行い在室者情報を可視化すれば、コミュニティ内で誰がいるのかを確認できる。これにより、コミュニティ内での交流や活動がスムーズになり、共同生活をする人たちが同じ時間にいる場合には、共同での食事や過ごし方を提案できる。また、コミュニティ内のイベントやミーティングの開催タイミングを調整できる。これにより、参加者が多い時間帯に開催できるため、参加者が集まりやすくなり、コミュニケーションの促進を行える。

また近年、在室者管理は新型コロナウイルスの影響により、感染拡大を防ぐ上で有効な手段と考えられている。在室管理システムを使用すれば、建物内の感染者を発見し、隔離できる。システムを使用すれば、感染者が訪れた場所や接触した人々を追跡し、消毒や清掃を行える。また感染が広がった可能性が高い人々を特定し、隔離や検査を行って早期発見や早期対応が可能になる。そのため、病院や医療機関などでは、在室管理システムを導入し、患者やスタッフの感染リスクを低減するために活用されている。

在室者管理の研究は、学術界や産業界においても注目を集めており、20世紀後半から様々な方法が提案してきた。在室管理を行う方法としてICカードを用いて在室者情報を能動的に記録する方法がある。例として、学生が教室に入るときにICカードをタッチし、出るときにも同様にタッチすることで、学生がどのくらいの時間教室に在室したかを記録できるシステムが挙げられる。ICカードの使用により、手動での在室情報の管理から自動化が可能になる。この方法では在室者情報を確実に記録できるがICカードを用いて能動的に記録する必要があるので利用者への負担がICカードの記録忘れによる在室情報の不正確さを招く可能性がある。近年では、様々なセンサを使用した方法が提案されている。例えば、照度センサ、温度センサ、音声センサ、カメラなどが挙げられる。これらセンサを使用して得られたデータを処理し、居住者がいるかどうかを判定する。ただし、これらのセンサを使用した在室者管理は、環境条件や居住者の生活スタイルなどにより精度が異なるため、正確な在室者管理ができない場合がある。このように、在室者管理は重要な課題であり、様々なアプリケーションに応用されるが、環境によって正確な検出が難しいという問題もある。また、深層学習を使用した、在室管理システムがある。深層学習は大量の画像や音声などのデータを特徴量として抽出し、学習を行うため、高精度の人物の在室検出が可能である。しかしながら、深層学習を使用した在室者管理には、大量のデータが必要であり、データ収集や学習には多くのリソースが必要なのが問題点である。また、プライバシ保護の観点から、カメラの使用は避けられる場合が多い。

我々の先行研究として BLE(Bluetooth Low Energy) ビーコン (以下、ビーコン) を用いた在室管理プラットフォーム「滞在ウォッチ」を提案している。ビーコンは低コストであり、普及しやすいと考えられる。これにより、建物全体に導入が容易であり、在室者検出のカバー範囲を広げられる。また、ビーコンは低消費電力であり、長期間にわたって使用できる。これにより、在室者検出を 24 時間行えるため、建物内での生活やビジネスに対して常時モニタリングを行える。さらに、ビーコンは個人を特定できないため、プライバシ保護にも適している。滞在ウォッチは、ビーコンを持ち歩き在室情報を記録するメンバ、現在状況や履歴を閲覧したり API を通して利用する利用者、メンバ管理、メンバへのビーコン配布、利用者の登録を行う管理者、システムを開発する開発者が存在する。メンバの負担軽減のため、在室者情報をビーコンで受動的に記録する方法を採用している。ビーコンを持ったメンバが部屋に訪れるとき受信機が検知し、サーバに在室者情報を送信し、データベースに記録する。データベースに保存された情報は、独自に作成した Web API によって外部からの利用が可能である。先行研究として、Web API を使用した退勤管理システムや在室情報可視化システム、部屋利用者の来訪促進システム、コミュニケーションプラットフォームなどがある。

1.2 滞在ウォッチの安定運用のためのシステム拡張

本研究の目的は、滞在ウォッチによる複数コミュニティ間連携の実現である。複数コミュニティ間とは空間的な距離が近いコミュニティ間と定義する。例として、大学の研究室間やビルのオフィス間などが挙げられる。滞在ウォッチにおける複数コミュニティ間連携によって得られる利点として、まずコミュニティ間でのコミュニケーションの増加が挙げられる。普段コミュニケーションを取らない別コミュニティの人々とのコミュニケーションが促進される。他のコミュニティから新しい知見やアイデアを共有による想像量や創造性の向上、スムーズなコミュニケーションやリソースを共有による、生産性が向上が期待できる。またコミュニティ間のコラボレーションが容易になるため、新しいプロジェクトやイニシアチブが生まれる可能性がある。

しかし滞在ウォッチを複数コミュニティ間で連携するには大きく分けて 2 つの問題点が存在する。まずコミュニティで独立した運用ができていない点が挙げられる。「独立した運用」とは、各コミュニティで自らのシステムや装置を運用できることを指す。複数コミュニティ間で連携するためには、各コミュニティで滞在ウォッチを運用するためには必要な設備やシステムを持つ必要がある。それぞれのコミュニティが独立したシステムを持っていれば、各コミュニティが独自に在室情報を収集し、管理できるため、それらを連携させられる。しかし既存の滞在ウォッチは單一コミュニティを前提として設計されているため各コミュニティが独立した運用を行えない。次に在室情報を長期に渡り継続的に記録できない点が挙げられる。在室情報の長期に渡る継続的な記録は、コミュニティ間でのコミュニケーションを促進するための基礎となる情報であるため必要である。しかし、既存の滞在ウォッチは BLE ビーコンのみを使用して運用しているため電池がなくなると在室情報を記録できず、電池の交換を促しているが交換がされず放置される場合がある。

これらの問題を解決した滞在ウォッチの運用を安定運用と定義する。本研究では滞在ウォッチを複数コミュニティ間で連携するために、独立したコミュニティにおける滞在ウォッチの安定運用のためのシステム拡張について提案する。安定運用に向けたアプローチとして既存システムの再構築、独立した運用に向けて利用者の管理とアクセス制御システムの整備、長期に渡る継続的な在室情報の記録に向けて BLE ビーコンのアプリケーション化とアプリケーション化を行ったスマホビーコンと実デバイスビーコンによるハイブリッドシステムの構築を行う

1.3 論文構成

第 2 章では、本研究と関連した研究との違いを比較する。第 3 章では、在室管理プラットフォーム「滞在ウォッチ」について述べる。第 4 章では、独立したコミュニティにおける滞在ウォッチの安定運用のた

めのシステム拡張について述べる。第5章では本研究に対するまとめと今後の課題について述べる。

第2章 関連研究

本章では、本研究と関連した研究との違いを比較する。2.1節では部屋における在室者検出方法、無線通信技術による検出手法の比較を述べる。2.2節では、部屋におけるスマートフォンやビーコン、ICカードを用いた在室者の検出方法の関連研究について述べる。2.3節では記録された在室者情報の提示方法の関連研究について述べる。2.4節ではコミュニケーション促進の関連研究について述べる。

表 2.1: 測位技術の比較

測位技術	屋外測位	屋内測位	消費電力
GPS	○	×	高い
携帯電話基地局	△	×	普通
Wi-Fi	△	○	普通
BLE	×	◎	低い

2.1 屋内位置推定における在室者の検出方法

以前から在室管理は自動化されれば便利なシステムになると言っていた[?]. 部屋における屋内位置推定にはいくつか在室者の検出方法があり、用途によって人の在否のみと個人を特定する方法がある。滞在ウォッチでは、利用者が目的とする人の居場所を把握できるように、個人を特定する方法に着目する。具体的にはICカードやライブカメラを用いた検出方法があり、これらには導入時に配線工事の手間や高価な機材を必要とするため困難であるとのプライバシーへの配慮が必要である。そこで、自動で在室者情報を記録する無線通信技術による検出方法に着目する。志毛らのBLEを用いた位置情報共有システムの開発[?]では、表2.1の無線通信による測位技術の比較を行った。消費電力が低く、屋内位置推定に向いているBLEを用いた検出方法が有効だと考えた。

2.2 在室者の検出方法に関する研究

大学や会社では在室者を検出する手法を用いて、講義の出欠[?][?][?][?][?]や勤怠管理[?]が行われている。在室者を検出する手法としてICカードを用いた検出方法[?][?][?][?][?][?]や、ビーコンの受信電波強度を利用した検出方法[?][?]がある。

ICカードを用いた在室者検出方法では利用者にICカードを携帯してもらい、専用の機器などを用いて在室者を検出する手法である。必要となる機器を導入した後は利用者はICカードを用いるだけで在室者検出ができる。しかし新たに導入する場合は必要となる機器や、ICカードを登録する手間などコストは高くなってしまう。

ビーコンの受信電波強度を用いた在室者検出方法には二つある。1つ目は利用者にビーコンを携帯してもらい、在室者を検出したい部屋に受信機を置く方法である。部屋利用者の在室者検出はビーコンの受信電波を受信機が取得するだけで行えるので、自動で行える。また、ビーコンはサイズが小さいものが多く、利用者が携帯する負担もからない。在室者検出する部屋に受信機を置き、利用者はビーコンを携帯するだけなのでコストも抑えられる。2つ目に利用者にはスマホを携帯してもらい、在室者を検出したい

部屋にビーコンを置く方法である。スマホの普及が進んでいる[?]ので、導入コストはビーコンのみであるため、1つ目の方法よりもコストを抑えられる。しかし、全員がスマホを所持しているわけではないので、スマホを所持していない利用者には別の検出方法を導入する必要がある。本研究では利用者が自発的に在室者情報を記録する手間を必要としない方法として、部屋ごとに受信機を設置し、個人がビーコンを携帯し自動で検出する方法を採用する。

在室者を検出し、在室者情報を管理するシステムに関する研究がある[?][?][?]. スマホを用いて在室者を検出し、在室者情報を管理するシステムは、スマホを所持している人が多いため、新たに必要となる機器の数が少なく、コストを抑えられる[?]. またビーコンを用いて在室者を検出し、在室者情報を管理するシステムは利用者にビーコンを携帯してもらい、在室者を検出したい部屋に受信機を設置すれば在室者を検出できるので、コストを抑えられる[?][?].

本研究ではスマートフォンを所持していない利用者も想定し、利用者一人一人にビーコンを携帯してもらい、在室者を検出したい部屋に受信機を設置する方法を採用した。

2.3 在室者状況の提示方法に関する研究

在室者を検出した後に在室者情報を用いた在室者状況の提示方法は様々ある[?] [?] [?] [?] [?]. 利用者全員が見られるサイネージに在室者情報を提示するものや、個人が所有しているスマートフォンやタブレットから在室状況を確認できるものなどがある。サイネージに提示する手法を用いた研究を図??[?], 図??[?]に示す。利用者全員が見られるサイネージに提示する手法では、一目で在室情報を確認できる必要がある。スマートフォンやタブレットに提示する手法を用いたものを図??に示す。また個人が所有しているスマートフォンやタブレットから在室状況を確認できる手法ではスマートフォンやタブレットに適するレイアウトを考える必要がある。本研究ではその場にいる人のコミュニケーション促進のために、その場にいる人が見られるサイネージに提示する手法を採用した。

2.4 コミュニケーション促進に関する研究

第三者間のコミュニケーションや知っている人同士のコミュニケーションを支援、促進する研究がある。まず見知らぬ他人や顔だけは知っているがコミュニケーションを取らない人とのコミュニケーションを支援促進するものがある[?][?][?][?]. 気軽に他者とのコミュニケーションができると示唆している。しかし知っている人同士のコミュニケーションを促進するものではなく、あくまで見知らぬ他者やコミュニケーションを取らない人とのコミュニケーションを支援、促進するものである。また公共空間でのコミュニケーションを支援、促進する研究がある[?][?][?][?]. 見知らぬ人や顔だけは知っている関係ではなく、同じ空間を共有している人同士のコミュニケーションを支援、促進するものである。コミュニケーションのきっかけや共有できる情報を提示するものとして、本研究でも同じ空間を共有している人同士のコミュニケーションを促進する方法として参考にする。

第3章 在室管理プラットフォーム 「滞在ウォッチ」

本章では、ビーコンを用いた在室者検出および在室管理プラットフォーム「滞在ウォッチ」について述べる。3.1節では、「滞在ウォッチ」のシステム概要、利用した機器やサーバ、ビーコンと個人の紐付け、在室者推定、在室者管理サーバを述べる。3.2節では、在室者情報の活用方法として、現在の在室状況を示す在室者情報のリスト、過去の在室履歴を示す滞在時間や曜日別の滞在率による可視化を述べる。3.3節では、滞在ウォッチによって記録された入退室時刻の評価について述べる。3.4節では、滞在ウォッチによって記録された在室者情報を用いた応用システムについて述べる。

3.1 「滞在ウォッチ」のシステム構成

滞在ウォッチは、ビーコンを持ち歩き在室情報を記録するメンバ、メンバ管理、メンバへのビーコン配布、利用者の登録を行う管理者、現在状況や履歴を閲覧したり API を通して利用する利用者（メンバや管理者も利用者になりうる）、システムを開発する開発者がおり、メンバがビーコンを携帯し、部屋ごとに設置された受信機によりビーコンを検出する手法で、在室者管理を自動に行う。メンバには一人1つビーコンを携帯してもらう。サーバには部屋利用者の名前とビーコンの ID を登録する。ビーコンは周囲に数秒に1回電波を発信する。受信機がビーコンの電波を受信する間隔は3分である。受信機が検出したビーコンの ID と電波強度はサーバに送信され、日時と在室した部屋名が記録される。記録した在室者情報は Web API を通して利用可能であり、勤怠管理システムや来訪促進システムといった様々な応用を想定している。

部屋の管理者と利用者に必要な情報を提供する「滞在ウォッチ」

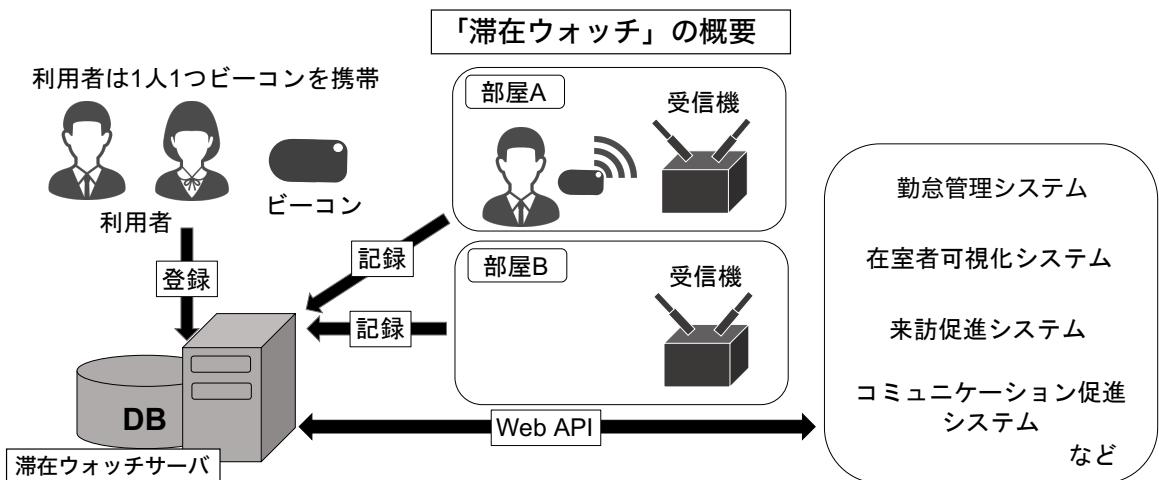


図 3.1: 「滞在ウォッチ」の概要図

本手法に関連するビーコン、受信機、サーバを示す。利用者が携帯するビーコンには、長期的な運用を考慮し電池交換が可能かつ小型な FCS1301[?] を利用している。ビーコンは図 3.2 のように様々な大きさ

や形のものがある。複数のビーコンの比較を表 3.1 に示す。ボタンは定期的な電波発信に加えて押した時にも電波発信できるため、点検時に利用できる。FCS1301 の用途は、財布やパスケースなどの貴重品に付ける紛失防止や子供の荷物などに付ける見守り支援がある。実際にパスケースに取り付けた様子を図 3.3 に示す。財布やパスケースに取り付けたり入れても気にならないサイズだとわかる。そのため、利用者が携帯するのに適している。また電池交換の際の様子を図 3.4 に示す。特殊な器具などを使う必要がなく、簡単に電池交換ができる。FCS1301 ではボタンを押すとペアリング、長押しするとスリープモードへ移行し、保管の間省電力モードになる。ビーコンの電波送信の間隔は FCS1301 の規格で最大の 10 秒ごとに設定している。部屋ごと設置する受信機には図 3.5 の低価格な Raspberry Pi[?] を利用している。1 つの部屋に 1 つずつ受信機を設置する。サーバには、Google Cloud Platform[?] を利用している。



図 3.2: ビーコンの種類

表 3.1: ビーコンの比較

ビーコン名	サイズ	電池交換	ボタン
FCS1301	縦 46.0 mm × 横 24.5 mm × 厚さ 3.5 mm	○	○
MAMORIO	縦 35.5 mm × 横 19.0 mm × 厚さ 3.4 mm	×	×
WICED	縦 60.0 mm × 横 37.0 mm × 厚さ 10.0 mm	○	○
estimote	縦 55.0 mm × 横 38.0 mm × 厚さ 15.0 mm	×	×



図 3.3: パスケースに取り付けたビーコン

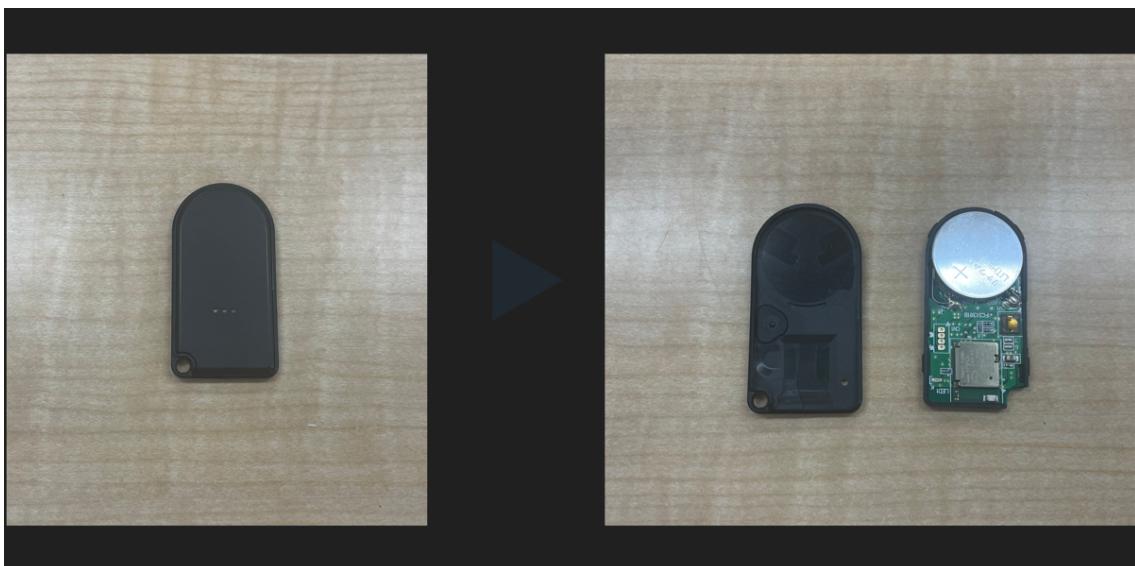


図 3.4: ビーコンの電池交換



図 3.5: 実際に使用しているビーコンを検知する受信機

個人を特定する在室者の検出手法には、個人と在室者情報を紐付けする必要がある。ビーコンを用いた検出手法ではビーコンのIDと利用者の名前をサーバのデータベースに登録している。

利用者がビーコンを携帯し部屋に訪れると、部屋ごとに設置された受信機でビーコンを検出する。この手法では、利用者は常にビーコンを所持している必要がある。FCS1301は小型で薄いビーコンのため、財布の中や鍵のキーホルダーとして貴重品などと一緒に持ち歩ける。この手法では、一時的に部屋を出た場合にも在室判定ができる、利用者はビーコンを所持するだけで、在室者情報を記録できる。利用者が部屋を訪れた際、部屋ごとに設置された受信機で検出されたビーコンの情報とサーバにある登録情報を参照し、在室者を特定する。また、複数の部屋に受信機を設置する可能性があるため、在室者の名前に加えて部屋名もサーバに送信する。部屋ごとに受信機を設置する際に、設置する部屋が隣接する場合に以下の問題が発生する可能性がある。ビーコンは周囲数メートルから数十メートルに電波を発信するため、隣接した複数の部屋で同様の在室判定を行う可能性がある。そこで、ビーコンが発信する電波強度に着目する。ビーコンの電波強度は表3.2に示すように壁や扉のコンクリートや鉄板の障害物によって減衰する[?].そのため、隣接した複数の部屋では電波強度に明らかな差があると考えられる。したがって、在室判定にビーコンの受信電波強度を利用して隣接した部屋での誤検出の防止が可能だと考える。

表 3.2: 高周波 (RF) 電波を反射／吸収する物質

障壁の種類	木材	合成物質	ガラス	水	煉瓦	大理石	土壁	コンクリート	金属
干渉の可能性	低	低	低	中	中	中	高	高	非常に高い

部屋ごとに設置された受信機によって検出された在室者情報はサーバに送られ、データベースに日時

と在室した部屋名が記録される。記録した在室者情報は Web API を通して利用可能であり、他のプログラムからも利用できる。Web API の利用方法は指定の URL を開くと JSON 形式 [?] で取得できる。例えば、<https://kajilab.net/stay-watch/stay> にアクセスすると現在の在室者情報が取得できる。実際に取得した JSON 形式の在室者情報を図 3.6 に示す。図 3.6 に示すように、現在在室している人の ID、名前、所属、在室している部屋を取得できる。この他にも <https://kajilab.net/stay-watch/> に続けてリクエストパラメータがある。リクエストパラメータについて表 3.3 に示す。

```

- {
  ID: "e7d61ea3f8dd49c88f2ff2484c07acb9-22823-42602",
  name: "suzaki",
  team: "ロケーション班",
  room: "学生部屋"
},
- {
  ID: "e7d61ea3f8dd49c88f2ff2484c07acb9-43319-20955",
  name: "hibi",
  team: "センシング班",
  room: "学生部屋"
}

```

図 3.6: JSON 形式で取得した現在の在室者情報

表 3.3: リクエストパラメータ一覧

パラメータ	値	説明
stay	string	現在の在室者を取得する
log	string	今日の在室情報のみを取得する
log?date1= YYYY-MM-DD &date2= MMMM-YY-DD	string	過去の在室情報を取得 (date1 から date2)
last-time	string	最後に滞在を確認できた時間を取得する
log-time	string	過去の累計滞在時間を取得する
log-time?month=YYYY-MM	string	過去の累計滞在時間を月ごとに取得する
log-group	string	その日だけのグループの滞在情報を取得する
log-group?date1=YYYY-MM-DD &date2=YYYY-MM-DD	string	期間を指定してグループの滞在情報を取得する (date1 から date2)

3.2 在室者情報の可視化

研究室やコワーキングスペースでは「今、誰がどこにいるのか」といった情報の共有が行われている。これは利用者において目的とする人の居場所が把握できれば、接触までのアプローチが容易になり、コミュニケーションの円滑化や共同作業を支援できる。管理者においても部屋の利用者数や時間帯が把握で

きれば、室内の温度調整を始めとする環境整備や活用状況が少ない部屋の省エネ化の指標となる。この在室者情報を確認できるように図3.7のリストを構築している。会いたい人の居場所を知るには、本人を探す、本人に連絡して聞く、事前に聞くといったアプローチが必要にある。本人を探す場合は、部屋が複数あったり、離れていたりするとそれは困難になる。本人に連絡して聞く場合は相手の状況、連絡手段によって、現在の情報を取得するには時間的な制限がある。事前に聞く場合、両者は時間的な拘束を受けており、いつでもコミュニケーションが取れるとは限らない。そのため、在室情報を何らかの方法で知れると、相手を介した居場所の確認の必要がなくなる。そこで、教員が部屋に在室しているかが外から把握できるように、図3.8に示すような教員の在室状況モニタを試験運用としてドアに設置した。この試験モニタによって、教員に部屋に在室しているか聞く必要がなくなり、アプローチしやすくなると考える。また、

滞在ウォッチ					
	Web上の名前	在室状況	在室部屋	最終更新日時	トータル
1	hamao	在室中	学生部屋	2019/1/18 9:54	59 pt
2	yoshimoto	-	-	2019/1/18 9:36	54 pt
3	Haruki.Inoue	在室中	学生部屋	2019/1/18 9:36	2257 pt
4	takai	在室中	学生部屋	2019/1/18 9:06	74 pt

図3.7: 在室情報のリスト



図3.8: 教員の在室状況モニタ

過去の在室状況からは利用者が人の訪問場所の傾向が把握できるため、次の人の訪問場所の予測がある程度可能である。これにより、円滑なコミュニケーションの補助になると考える。例えば、相手に会う約束をするほどではないが、一緒に作業をしたら効率が良くなるといった場合が該当する。管理者は部屋の利用状況を把握でき、室内の温度調整を始めとする環境整備や活用状況が少ない部屋の省エネ化の指標となる。そこで過去の在室履歴を図??の滞在時間、図??の曜日別の滞在率、図??の班別の滞在率、図??の合計滞在時間をグラフで示している。

3.3 部屋利用者の入退室時刻の評価実験

滞在ウォッチで記録された部屋利用者の入退室時刻が正確に記録されているか、ビーコンが電池切れをせずに記録されているかを確認するための評価実験を行う。滞在ウォッチのシステム構成でも説明した通り、受信機がビーコンの電波を受信する間隔は3分である。そのため、滞在ウォッチで記録される入退室時刻は実際に入退室した時刻と比べて誤差が3分以内であるのが望ましい。また在室管理プラットフォームであるため、来訪した部屋利用者全員の在室者情報が記録されてなければいけない。電池切れをせずに正確に記録されているか、電池切れをしている利用者がどれくらいいるかを見る。

本研究で使用しているビーコンは、電池で動作しているため、電池切れの際は電池交換をする必要がある。電池が切れていると在室者情報は記録できないため、ビーコンの電池が切れているかどうかを通知する必要がある。通知をする手段としてSlackでの通知を行う。実際に電池切れを通知した際の様子を図3.9に示す。Slackでの通知を行う理由として、研究室に所属する人全員がSlackを通して連絡やコミュニケーションをとっているからである。電池切れ通知は、5日以上受信機がビーコンの電波を検知できない人に対する。5日以上の理由として、本研究室では火曜日と金曜日にミーティングが行われるため、5日以上間隔が開く場合は電池切れの可能性が高いからである。しかし図3.9のように、100日以上検知できていないにも関わらず、電池交換がされていない利用者がいる。電池交換をしていない利用者の特徴として、研究室にこないため電池交換をそもそもしない、通知が来ても電池交換が面倒でそのままにするのが挙げられる。そのため電池切れの通知だけでは電池交換を行ってくれない可能性がある。また、Slackでの通知のみでは利用者がメッセージの重要度を低く見積もる可能性があるため、管理者からも利用者に対し適宜コンタクトを取る必要があると考える。電池切れの通知による電池交換の実行率や完了率を増やすには、改善点として通知頻度の見直しが挙げられる。

 stay-watch アプリ 09:50
電池切れの通知もしくは5日以上部屋に入室したデータがない人への通知です。
電池交換の手順は<https://kjlb.esa.io/posts/1996>で確認しながら行ってください。

対象ユーザ

最終確認日からの経過時間(5日以上の人のみ)

- @sawai: 6.3日
- @matsubara: 59.3日
- @yoshimoto: 265.3日
- @maruyama: 16.3日
- @iwao: 93.1日
- @shamo: 135.4日
- @akito: 7.3日
- @rui: 14.3日
- @fueta: 142.4日
- @suzaki: 19.2日
- @muneyasu: 16.3日

図 3.9: Slack による電池切れ通知の実際の様子

2022年1月11日に入退室した13人の部屋利用者と時刻をビデオで撮影し、実際に入退室した時刻と滞在ウォッチで記録された時刻を比較する。比較した結果を表3.4、表3.5に示す。比較結果として入室時刻の平均誤差が1分、退室時刻の平均誤差が2分であった。3分毎に受信機がビーコンの電波を検知するので、3分以内の誤差であれば十分な精度であると考える。しかし、来訪したにも関わらず、記録されていない利用者もいた。今回在室者情報が記録された利用者は13人、記録できなかった利用者は7人であった。つまり、記録できなかった利用者は電池切れの通知をしても電池交換をしなかったと考えられる。今後の課題として、在室者情報が記録できなかった利用者に対してアンケートを行い、なぜ電池交換をしないのか、どのような電池切れ通知するべきか、通知方法、通知頻度を聞く必要がある。

表 3.4: 実際の入退室時刻と滞在ウォッチで記録された時刻の比較

	実際に入退室した時刻		滞在ウォッチで記録された入退室時刻	
	入室時刻	退室時刻	入室時刻	退室時刻
利用者 A	09:01	10:51	09:03:15	10:53:13
利用者 B	08:43	10:27	08:44:56	10:31:50
利用者 C	08:25	10:27	08:26:37	10:31:50
利用者 D	10:20	19:43	10:22:40	19:46:05
利用者 E	10:46	14:51	10:47:06	14:53:59
利用者 F	10:43	13:31	10:44:03	13:34:54
利用者 G	10:46	15:30	10:47:06	15:33:30
利用者 H	10:52	12:43	10:53:13	12:46:12
利用者 I	10:25	14:40	10:25:43	14:41:48
利用者 J	10:05	15:26	10:07:24	15:30:28
利用者 K	12:05	18:25	12:06:30	18:26:58
利用者 L	10:28	14:29	10:28:46	14:32:41
利用者 M	12:05	17:26	12:06:30	17:29:11

表 3.5: 実際の入退室時刻と「滞在ウォッチ」で記録された時刻の平均絶対誤差

入室した時刻の平均絶対誤差	退室した時刻の平均絶対誤差
1分	2分

3.4 在室者情報を用いた応用システムについて

部屋利用者の在室者情報を用いれば、部屋の管理者や部屋利用者にとって便利な応用システムを考えられる。1つ目の応用システムとして、勤怠管理システムが考えられる。勤怠管理システムの代表例としてタイムカードでの打刻や、ICカードやQRコード、スマートフォンによる記録が挙げられる。これらのシステムでは利用者が能動的に記録する必要がある。本研究でのビーコンを用いた方法では能動的な記録を必要とせず、自動で記録されるため、利用者の負担や記録のし忘れが減ると考えられる。2つ目の応用システムとして、コミュニケーション促進システムが考えられる。研究室では部屋利用者同士の活発なコミュニケーションが求められる。しかし研究室に所属する部屋利用者は必ずしも共通の話題を持つ人が集まるわけではない。そこで部屋利用者が共有できる情報である在室者情報を用いてコミュニケーションを促進するようなシステムを構築する。コミュニケーション促進システムについては4章で実装した「きょうの滞在」で詳しく述べる。3つ目の応用システムとして、来訪促進システムが考えられる。在室者情報を用いたゲーミフィケーションに基づくシステムが考えられる。具体例として在室履歴を用いたペット育成ゲームが考えられる。在室履歴を用いたペット育成ゲームの概要を図3.10に示す。部屋利用者の在室履歴からペットに餌を与えるようなシステムである。部屋利用者はペットに餌を与えるためには部屋に在室する必要がある。そのためペットに愛着があるなら部屋利用者は来訪する必要がある。また在室履歴を用いたペット育成ゲームのアプリのイメージ図を図3.11に示す。誰でも見やすいようなレイアウトやスマホでの利用を想定したボタン配置を意識した。

ある滞在条件を満たすとペットの満腹度効率が向上

来訪頻度	同室人数	組み合わせ	満腹度
多い 	多い 	珍しいグループ A D	 増えやすい
少ない 	少ない 	いつものグループ A B C	 増えにくい
曜日によって来訪する人の偏りをなくす		来訪する人の固定化を防ぐ	

図 3.10: 来訪促進システムの概要図



図 3.11: ペット育成ゲームのアプリのイメージ

第4章 独立したコミュニティにおける滞在ウォッチの安定運用のためのシステム拡張

滞在ウォッチを複数コミュニティ間で運用するには、独立したコミュニティで運用できない、長期に渡り在室情報を継続的に記録できない問題を解決した安定運用を行う必要があり、それに向けた取り組みについて述べる。4.1章で既存の滞在ウォッチに独立運用するためのシステムの再構築について述べる4.2章で利用者の管理とアクセス制御システムの整備について述べる4.3章でスマホアプリによるビーコンと実デバイスの併用について述べる。

4.1 安定運用に向けた既存システムの再構築

既存の滞在ウォッチは長期に渡り継続的に保守・開発していくのが困難だったため安定運用を目指す上でシステムの再構築を行った。

4.1.1 サーバの再設計と再構築

サーバ側には独立したバックエンドシステムとの連携が容易でありWebアプリケーションがより高い可用性とスケーラビリティを実現できるREST APIを採用した。REST APIは、複数のクライアントからアクセスができるため、様々なデバイスやプラットフォームからアクセスできる。これにより、より広いユーザー層からアクセスが期待できる。

滞在ウォッチのサーバ側システムはPythonで構築されていたが、高負荷環境でパフォーマンスに影響を与える可能性があったのと動的型付け言語の開発、保守のしづらさを解消するため、Go言語を使用してシステムの再構築を行った。既存の滞在ウォッチのサーバ側のシステムはPythonを用いて構築されていた。Web APIは大量の処理をする必要があり、Pythonの実行速度の遅さが、高負荷環境でパフォーマンスに影響を与える可能性がある。またデフォルトで非同期プログラミングをサポートしていないため、非同期処理に対応するために追加のライブラリを使用する必要がある。さらに動的型付け言語なため、型の不明確さによってバグを見つけづらく保守性が低い。そこで静的型付けであり、高速な処理能力と小さなメモリフットプリントを持つため、Web APIの開発に適しているGo言語を採用した。Go言語は高いパフォーマンスを持つ。C言語のような低レベルな言語と同等のパフォーマンスを持ちながら、高水準の言語のような簡単な構文を持っているのに加え、並列処理を容易に実現できるため、高負荷な環境でのWeb APIの開発にも適している。そのため、WebAPIの要求に適合するために必要なパフォーマンスを提供できる。また構造体とインターフェース型を備えており、WebAPIの開発に必要な柔軟性を持っている。これにより、開発者は、WebAPIを実装するために適した方法を選択できる。さらに、標準パッケージによるHTTPサーバのサポートを持つため、Web APIの開発に必要な機能を簡単に実装できる。

サーバで使用するデータベースには無料で利用でき、高いスケーラビリティを持つため、WebAPIの要求に対応可能なMySQLを採用した。MySQLは多くのプラットフォームでサポートされており、多くの言語に対応しており、開発者にとって選択肢が広がる。またMySQLは優れたセキュリティ機能を持つため、WebAPIのセキュリティの確保ができる。

Go言語でMySQLの操作を行う際Go言語のORMライブラリであるGormを採用した。ORMはデータベースを操作を行うための手法の一つである。これにより、データベース操作を行う際に、SQL文を直

接記述する必要がなくなり、コードがすっきりし、可読性が高くなる。またSQL分を直接記述しないためSQLインジェクション対策にもなる。SQLインジェクションとは不正なSQL文の挿入を行い、データベースに対して攻撃を行う手法である。ORMは、プログラマが入力したデータを自動的にエスコープし、サイバー攻撃につながるような文字を無効化するためSQLインジェクション攻撃を防止できる。GormはGo言語向けORMライブラリの一つである。Gormはデータベースの挿入、更新、削除、検索などの基本的な機能操作をサポートしている。また自動的なマイグレーション機能を持っている。これにより開発者はデータベーススキーマを手動で管理せずに、スキーマの変更を自動的に反映できる。GoのORMライブラリとして他にXormなどの選択肢もあったが複雑なクエリを簡単に作成するための豊富なクエリビルダーAPIを提供しているためGormを選択した。

サーバサイドプログラムの開発及びデプロイメントする際にはプログラムを実行するために必要な全ての環境をコンテナ内にパッケージを行い、開発環境と本番環境での環境差を最小限に抑えられるDockerを採用した。Dockerを使用すれば、アプリケーションを実行するために必要な全ての依存関係を詰め込んだイメージを作成できる。これにより、開発者は、アプリケーションを実行するために必要な環境を1つのイメージにまとめられ、それを任意の環境に簡単に展開でき開発環境の構築が容易になる。さらにDockerは他のアプリケーションの影響を受けずに実行可能である。これにより、アプリケーションが依存しているライブラリやフレームワークのバージョンなどの環境の違いによって、アプリケーションが動作しないといった問題を回避できる。またコンテナを水平にスケールアウトできるため、アクセスが集中した場合にも、スムーズな処理を行える。デプロイ先は、研究室内のサーバを利用した。研究室のサーバの利用はクラウドサービスに比べて、長期的にコストを削減できる。

4.1.2 クライアントサイドの実装

クライアントサイドでは、WebAPIを利用して、Webアプリケーションを実装した。Webアプリケーションのフレームワークとして、Next.jsを採用した。Next.jsはReact.jsをベースにしており、React.jsの開発を効率化に加え、サーバーサイドによるレンダリングをサポートによるSEO対策やパフォーマンスの向上を実現できる。React.jsはJavaScriptのライブラリであり、コンポーネントベースのアーキテクチャを採用している。コンポーネントを独立した単位として開発すると、開発者はそれぞれのコンポーネントに集中できるため、開発効率が向上する。コンポーネントは再利用可能な単位であり、複数のページで同じコンポーネントを利用することができ、これにより、開発コストを削減し、保守性を向上させる。またReactは、仮想DOMとは、実際のDOM(Document Object Model)を操作する代わりに、JavaScriptのオブジェクトとして扱う仕組みを採用している。仮想DOMは、実際のDOMと比較して、更新が必要な部分だけを計算を行うため、パフォーマンスの向上が図れる。

4.2 利用者の管理とアクセス制御システムの整備

独立した運用を行うためには利用者の管理とアクセス制御システムの整備が必要である。既存の滞在ウォッチは単独コミュニティでのみの運用を前提としており開発者と管理者が同一であった。Web上からユーザの登録をするシステムが存在せずデータベースに対して直接変更を行うSQLを発行していた。單一コミュニティのみでの運用であったため成り立っていたが、複数コミュニティ間で運用を行う場合、コミュニティの数が増えるに連れてシステム開発者の利用者管理の負担が大きくなり運用するのは難しくなる。また在室情報はプライバシに関わるものであるため、外部のものがWebページにアクセスしても在室情報を閲覧できないようにする必要があるにも関わらず、誰でも閲覧可能な状態であった。これらを解決するには利用者のアクセス制御システムの整備とコミュニティごとの管理者を作り、各コミュニティで独立した運用を行う必要がある。各コミュニティごとに管理者が存在すればコミュニティの利用者の管理を開発者が全て行う必要がないため負担が軽減される。

そこで Firebase Authentication の Google プロバイダーによる OAuth 認証を実装した。 OAuth (Open Authorization) は、ユーザーがサービスプロバイダー（例えば Google, FaceBook など）に対して、別のサービスにアクセスするための権限を与えるためのオープンスタンダードの認証プロトコルである。 OAuth 以外の認証方式として Basic 認証などがある。 Basic 認証はユーザ名とパスワードをベースック認証ヘッダーに埋め込んで送信する方式である。この方式はセキュリティ上の問題が存在する。パスワードが暗号化されていないため、パスワードが盗まれる危険がある。また、Basic 認証はバックエンド側でユーザ名とパスワードを保存する必要があり、管理を行うにはパスワードのハッシュ化など適切な処理を施す必要がある。それと比較して OAuth 認証では、ユーザのアカウント情報を第三者にアプリケーションに渡さずにアクセス権を付与できるため、セキュリティ上の利点がある。 OAuth 認証では、アクセス権を持つアプリケーションにのみ有効で、期限切れになると、使用できなくなる。これにより、アクセス権を付与したアプリケーションが不正にアクセスするのを防止できる。これらの理由により、OAuth 認証を採用した。 OAuth プロバイダーには Apple, FaceBook など複数の選択肢がある中で Google を採用した。 Google アカウントは世界中で広く利用されており、多くの人がアカウントを持っており、多数の利用者に対応可能である。

Google アカウントを用いた OAuth 認証は、Google アカウントを使って、サービスプロバイダー (Google) に対して、利用者が滞在ウォッチシステムにアクセスするための権限を与える。利用者は Google アカウントにログインし、Google が提供する OAuth 認証プロセスを通じて、滞在ウォッチシステムにアクセスできる。 OAuth 認証により、利用者は自分の Google アカウントの情報（パスワードなど）を滞在ウォッチシステムに入力をせずに、安全にアクセスできる。しかし Google アカウントによる OAuth 認証だけでは意図しない外部のユーザのアクセスを防ぐことはできない。なぜなら Google アカウントを持っているユーザなら誰でも認証を行えるためである。そこで Google アカウントによる OAuth 認証に加えて滞在ウォッチサーバ側で独自のユーザ認証を行った。具体的には滞在ウォッチ管理者が登録したユーザのみを認証するというものである。滞在ウォッチサーバのデータベースには Google のアカウントのユーザ名であるメールアドレスが保存してある。この保存されたメールアドレスは滞在ウォッチの管理者が登録したものであるため、メールアドレスが存在する場合は管理者の許可ある正規のユーザと判別できる。

管理者のユーザ登録フロー図を図 4.1 に示し具体的に説明する。まず初めに利用者は管理者に対して Google アカウントを報告する。報告後管理者が Web の管理者ページのユーザ登録画面から Google アカウントを登録する。ユーザには 2 種類のパターンが想定される。まず 1 つ目は BLE ビーコンが既に登録済みのユーザ場合の登録である。図 4.2 にその登録画面を示す。

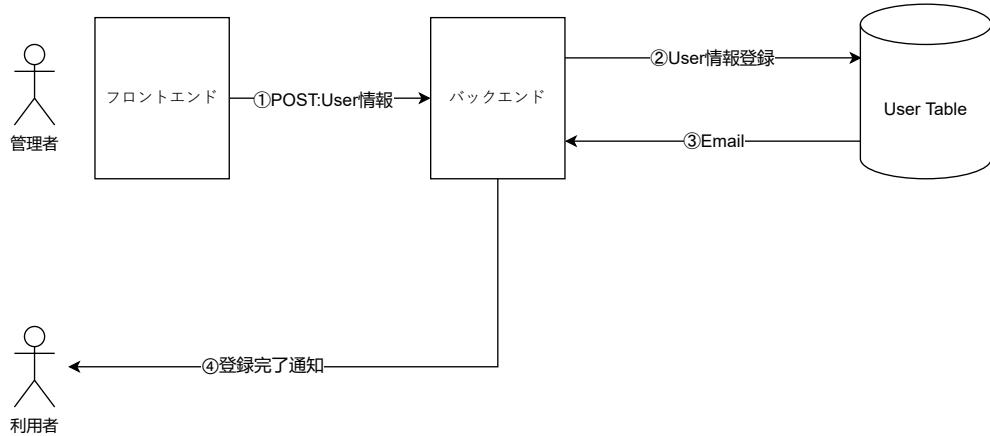


図 4.1: 認証システム：管理者側のユーザ登録フロー図

この登録画面は以前から滞在ウォッチユーザとして登録がされておりビーコンに関する情報とユーザ情

報が既に存在しており Google アカウントの情報とユーザロールのみがないユーザを想定している。ユーザネームはデータベースに存在しているためバックエンドからデータを取得できる。この登録画面は、以前から「滞在ウォッチャユーザ」として登録されているユーザ向けで、ビーコンに関する情報やユーザネームの情報は既に存在しており、Google アカウントの情報やユーザロールのみがないユーザを対象としています。取得した利用者のユーザネームデータを管理者が選択できるようにセレクトボックスを配置した。ユーザロールのところでユーザの権限レベルの指定ができる。一般ユーザと管理者ユーザの2つがあり、一般ユーザは滞在ウォッチの在室情報の閲覧などを行え、管理者ユーザはユーザの登録を行える。管理者が登録ボタンを押すと、POST リクエストがサーバ側に送信される。登録されると管理者が選択したデータベースに存在するユーザにメールアドレスが紐づき、サーバ側は対象のメールアドレスに対して登録完了のメールを送信する。メールアドレスに対して通知を行うのはユーザがいつ登録が完了したのか気づけるようにするためである。2つ目に想定されるのが BLE ビーコンが登録されていない新規ユーザである。登録画面を図 4.3 に示す。ユーザは 4.3 章で説明するスマホビーコンを使用する新規ユーザである。新規ユーザにはユーザネームがないため管理者はユーザネームを入力する必要がある。その他は先に述べた登録画面と同じであり、登録ボタンを押しサーバに対してリクエストを送信する。先に述べたものとの違いはサーバ側は新規の UUID の生成を行うところである。スマホビーコンの初期セットアップ時に UUID の設定を行うが、その際ユーザの Google アカウントに対応した UUID をセットアップできる。

メンバーの招待

BLE ビーコン登録済み

BLE ビーコン未登録

ユーザ選択 *

ユーザを選択

Gmailアドレス *

your@gmail.com

ユーザロール *

一般ユーザ

登録する

図 4.2: BLE ビーコン登録済みの登録画面

メンバーの招待

The screenshot shows a registration form titled 'メンバーハウスの招待' (Member Invitation). At the top, there are two tabs: 'BLEビーコン登録済み' (BLE Beacon Registered) and 'BLEビーコン未登録' (BLE Beacon Unregistered), with the latter being active. The form contains three input fields: 'ユーザネーム *' (User Name *) with placeholder 'your name', 'Gmailアドレス *' (Gmail Address *) with placeholder 'your@gmail.com', and 'ユーザロール *' (User Role *) with a dropdown menu currently showing '一般ユーザ' (General User). A large blue button at the bottom right says '登録する' (Register).

図 4.3: BLE ビーコン未登録の登録画面

次にユーザ側の認証のフローを説明する。Fireabase Auth で認証が成功すると JWT(JSON Web Token) トークンが発行される。Firebase Auth の JWT トークンは Firebase Auth が認証済みのユーザーを確認するために使用するトークンである。JWT トークンは、JSON 形式の文字列では発行者、トークンの有効期限、トークンの使用目的、サブジェクトが含まれておりユーザを一意に識別可能である。JWT トークンは、フロントエンドから滞在ウォッチのバックエンドに送信される。滞在ウォッチのバックエンドは Firebase Auth の API を使用して、JWT トークンの検証を行い、トークンが有効であるか確認する。有効な JWT トークンであった場合、アプリケーションのバックエンドは、そのトークンに含まれる情報を使用して、Firebase Auth が認証済みであることを確認する。認証済みであった場合は次にメールアドレス情報を確認する。データベースにメールアドレスが存在する場合フロントエンドに対してアクセスの許可を行う Response Code 200 を返す。このプロセスによって不正なユーザの在室情報の閲覧を防いでいる。よって適切な範囲での在室情報の共有が可能である。

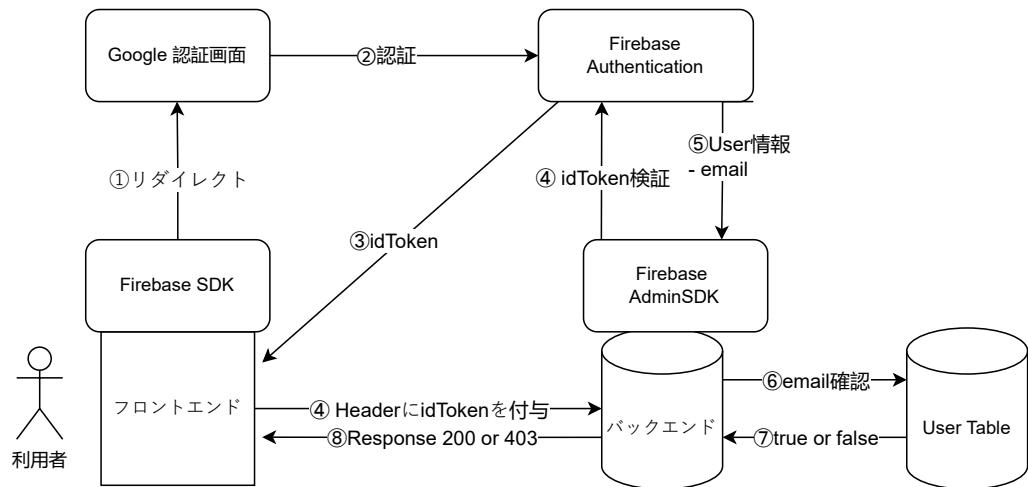


図 4.4: 認証システム: ユーザ側のフロー図

4.3 スマホアプリによるビーコンと実デバイス

我々の利用するシステム「滞在ウォッチ」は実デバイスによるビーコンを利用した、ここではビーコンのアプリケーション化及びそれらのハイブリッド活用について記す。

「滞在ウォッチ」は、先述の通り BLE ビーコンを利用している。初めに BLE ビーコンの仕様について解説する。BLE ビーコンとは BLE 規格のペリフェラル（Peripheral）動作を用いて、周囲にアドバタイズを行うデバイスとセントラル（Central）動作を用いて周囲をスキャンするデバイスによって構成される。ペリフェラル動作とは通信を受け付ける子機としての動作である。セントラル動作によってスキャンしたデバイスに反応し GATT（Generic attribute profile）内に保持したサービスやキャラクタリストリックのデータの送受信を行う。キャラクタリストリックとは、そのデバイスが保持するサービスやデータであり、今回の研究では個人識別符号として利用する UUID が該当する。UUID（Universally Unique Identifier）とは、オブジェクトを一位に識別するために重複がない前提で用いる 128bit の数値である。キャラクタリストリックのデータとしてこれをビーコンは保持している。

実デバイスによる BLE ビーコンは低コストでありサイズもコンパクトと携帯が用意である。しかしながら、メンバや管理者にとって利便性が低く結果として可用性が低くなる。可用性とは、メンバの在室情報が長期にわたり継続的に記録される能力と定義する。利便性における問題として、バッテリ切れによる問題、初期設定が複雑である問題が存在する。まずはバッテリ切れによる問題である我々が利用しているビーコンはバッテリとして CR2016 を利用している。この CR2016 は一般的なコイン型リチウム電池であり、また実デバイスによるビーコンにバッテリ切れを通知する機能はない。バッテリ残量の把握には、ビーコンメーカーによる指定のアプリケーションを要し、実デバイスによるビーコン一台ごとに接続する必要がある。また納入されたビーコン FCS1301 はバッテリ切れによるデータ初期化が発生しない点を重視し購入したが、我々がシステム運用をしている際にバッテリ切れが発生していない状態でも何故か UUID が設定したものと異なる数値に変更されてしまうケースが散見された。

他にも滞在ウォッチに特化したビーコン設定アプリを作ろうとした際、メーカーにビーコン設定用アプリケーションに用いているライブラリについて相談したところ「メーカーの専用アプリを利用してください」と期待通りの返答は得られなかった。これらの状況から、実デバイスによるビーコンのみの運用では、メンバにとっての利便性が低下してしまい、それが原因としてシステムの可用性が低下してしまうと判断した。

4.3.1 BLE ビーコンのアプリケーション化

上記の問題のアプローチとしてメンバの利便性を向上させるため、スマホアプリによるビーコン動作を行った。

BLE ビーコンの代替としてスマートフォンを利用可能にするとバッテリ交換の手間が削減される。またスマートフォンユーザーにとってスマートフォンはコミュニケーションツールとしての用途からバッテリ切れを配慮する傾向が強い。実デバイスによるビーコンと比べてスマホビーコンはバッテリが維持されやすく利便性が向上すると考えた。実デバイスによるビーコンには、バッテリ残量やバッテリ切れを通知するユーザインターフェースを持たない。しかし、スマートフォンは液晶画面を持っており、スマートフォン自体のバッテリ残量やスマホアプリによるビーコンの動作状況など可視化が可能である。そのためビーコンの動作状況を表示することでビーコン動作の停止に気が付きやすく、メンバによる再起動が行われた場合、可用性の向上が期待できる。これらのメリットからスマートフォンアプリによるビーコン動作がユーザーの利便性を向上させ、システムの可用性へつながると考えた。



図 4.5: 通知領域におけるビーコンの動作状況の表示

スマホビーコンは基本的にバックグラウンドに常在させる利用法を想定し実装した。既存研究ではメンバに実デバイスによるビーコンを携帯させ、能動的な記録動作の必要がない。バックグラウンドに常在させる方式は実デバイスによるビーコンと同等の利便性がある。そのためバックグラウンドへの常在がアプリケーション化の前提である。その点を踏まえて動作プラットフォームを選定した。選定理由は技術調査をした際に、現状の iOS ではフォアグラウンド動作はするもののバックグラウンド動作に制限がかかっており実デバイスによるビーコンと同等の利便性を担保することが困難であると判明したため、対応プラットフォームを Android のみにした。

実装に伴い、アプリケーションのプログラムは Kotlin を用い、Bluetooth 関連の実装においては AltBeacon というライブラリを利用した。また、ビーコン動作をする上で、Android のプライバシ規則によって、バックグラウンドでのユーザーへの通知無しでの動作は禁止されている。その点と、先述のスマートフォンが表示機能を持つ点を利用して 図 X に示す通り 通知領域上で ビーコン動作中は常時表示する仕様とした。この仕様によってユーザーが入退室のたびにアプリの起動を行う必要もなく、プライバシ規則を遵守した上で、実デバイスによるビーコンと同様の能動的な記録動作を要しない利便性を担保した。

それに伴い、管理者が実デバイスによるビーコンで行っていた登録作業をより簡略化した。従来では管理者が新しいメンバが増えるたびに登録作業をおこない、メールアドレス、名前、UUIDなどを登録し、それに合わせて実デバイスによるビーコンへ割り当てられた UUID を登録する作業を行っていた。しかし先述の通り、Firebase によるログインによって、スマホビーコン上で利用する UUID をメンバに紐づいているメールアドレスでログインすることで、利用できるようにした。

また図 X 研究初期はユーザーから利用している UUID,major,minor を編集できる仕様にしていたが、複雑な操作がユーザーの利便性の向上につながらないと考え取り消した。

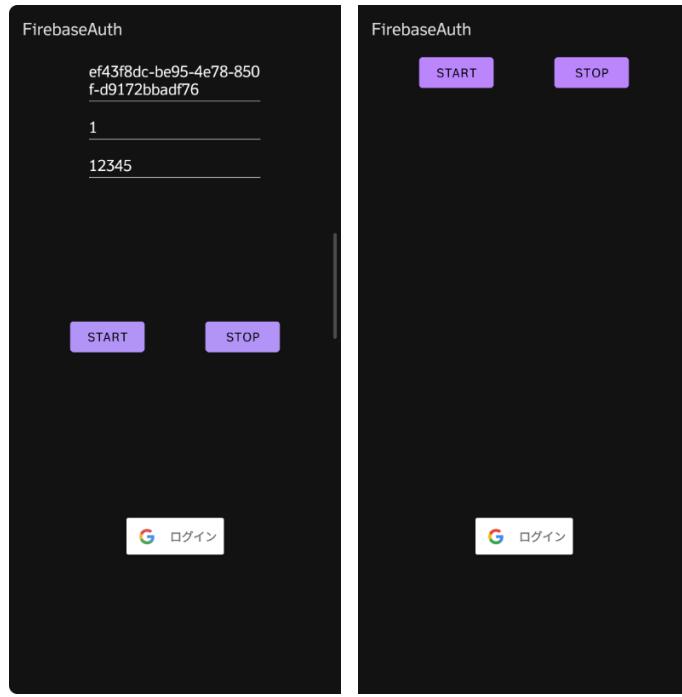


図 4.6: メンバ向けアプリケーションにおける操作の簡略化

4.3.2 スマホビーコンと実デバイスによるビーコンのハイブリッド活用

スマホビーコンのみを利用する場合、様々な状況下でメンバの継続した利用が困難であるため、実デバイスによるビーコンと併用できるシステムとした。長期にわたり継続的に利用するメンバにとっては、実デバイスによるビーコンは先述の通りバッテリ交換の手間がある。スマホビーコンはそのようなメンバにとっては、バッテリ交換の手間がないため有用である。しかしスマートフォンを携帯していないメンバ、スマホビーコンの利用に伴うバッテリ消費が気になるメンバなども想定される。これらの問題は実デバイスによるビーコンとスマホビーコンのハイブリッド化によって解決できる。スマホビーコンで利用するUUIDを実デバイスによるビーコンで利用するUUIDと同じ値に設定し同じメンバの在室情報を記録している。この方法は表 X に示す通りスマホビーコンか実デバイスによるビーコンの少なくとも片方を携帯していれば記録できるため継続的にデータを記録する観点から見ても有用である。

第5章 おわりに

5.1 まとめ

今回 BLE ビーコンを用いた在室者検出システム「滞在ウォッチ」を複数間コミュニティで連携するためのシステム拡張を行なった。滞在ウォッチは BLE ビーコンをメンバに持たせて、自動的に行うシステムである。滞在ウォッチを複数間コミュニティで連携するには、独立したコミュニティで運用できない、長期に渡り在室情報を継続的に記録できない問題を解決した安定運用を行う必要がある。独立したコミュニティの運用へのアプローチとして利用者の管理とアクセス制御システムの整備を行なった。その結果、スケーラビリティの高い利用者の管理と適切な範囲での在室情報の共有が可能となった。長期に渡り在室情報を継続的に記録できない問題へのアプローチとしてスマホアプリによるビーコン化と実デバイスの併用を行なった。その結果、実デバイスのみで在室管理していた時と比べて、長期に渡り在室情報を記録できるようになった。

5.2 今後の課題

今後の課題としてまず BLE ビーコンによる在室判定の誤判定が発生する問題がある。例えば受信機の設置してある部屋の上の階の部屋にいる人が入室したと判定される場合がある。この問題への対応策として、信号強度を利用する方法などがある。信号強度を利用する方法は、受信機が受け取ったビーコンの信号強度を測定し、その部屋にいるかどうかを判定する方法である。信号強度は、送信機から受信機までの距離に比例して減衰する。そのため、受信機が受け取ったビーコンの信号強度が弱い場合は、その部屋にいる可能性が低いと判断できる。具体的には、受信機が受け取ったビーコンの信号強度を測定し、それが一定のレベル以上であれば、その部屋にいると判断し、そうでなければ、その部屋にいないと判断する。ただしこの方法は、受信機が受け取ったビーコンの信号強度を基にして、その部屋にいるかどうかを判断するため、部屋の形状や構造によっては、誤った判断をする可能性があり、全ての部屋で正しく判定するには、部屋ごとに応じた閾値を設定する必要があり、コストがかかる。他の方法としては、BLE の以外の通信規格を使用する方法が考えられる。Ultra-Wideband(UWB) デバイスや RFID デバイスを使用した方法がある。UWB デバイスは、壁を貫通しにくく、高精度な位置測定を行うことができるが、新しい技術であるため、普及が送れており、対応するデバイスやアプリケーションが少ない欠点がある。Zigbee デバイスは、ワイヤレス通信技術の一種で、壁を貫通しにくい低功率の電波を使用し範囲が狭いため、隣接する部屋に影響を与えるのが、伝達速度が遅いため、大量のデータを伝送する場合には向いていない欠点がある。UWB デバイスや Zigbee デバイスを使用した方法は上記で述べた欠点もあるが部屋を貫通しにくく、誤判定が少なると考えられるため、導入を検討する価値があると考える。次に、BLE ビーコンの UUID を設定する手間がかかる問題がある。現在 UUID をセットアップする際には、専用のアプリを使って UUID の書き込みを行う必要がある。この問題への解決策として、UUID を書き込むアプリを独自に作成する方法が考えられる。独自のアプリを作成すれば滞在ウォッチサーバ側に対して HTTP リクエストを送りそのレスポンスとして UUID 取得してそれを BLE ビーコンに書き込むようなシステムを構築できると考えられる。BLE ビーコンの提供先である株式会社フォーカスシステムズに対してビーコンへの書き込みを行う API を提供してもらえないか確認を行ったが、結果としてそのような API は提供していないとの回答であった。しかし専用のアプリで BLE ビーコンへの UUID の書き込みを行なっているため不可能ではないと考えられる。そのため BLE ビーコンのセットアップの設定の簡略化を行うため

に、4.3章で話したスマホビーコンの機能としてBLEビーコンのUUIDを提供する機能の追加を行なっていきたい。

プロジェクトの課題として、滞在ウォッチの複数コミュニティ間連携を実際に用いて運用を行う必要がある。本研究では、滞在ウォッチが複数コミュニティ間で連携できるように独立したコミュニティにおいて滞在ウォッチの運用ができるように検討を行ったが、実際に複数間コミュニティ間で運用後に発生する問題も考えれる。そのため、運用後はメンバ、管理者、利用者からの意見や得られたデータを元に評価、システムの改善を行う予定である。また現状のシステムでは複数コミュニティ間でのコミュニケーションを促進する機能の実装には至っていない。そのため複数コミュニティに向けたイベント開催機能などを実装し運用・評価する必要がある。

謝辞の例

本研究を進めるにあたり、多くの御指導、御鞭撻を賜わりました愛知工大教授に深く感謝致します。

また、御討論、御助言していただきました、○×大学工学部電子情報工学科の山谷川介教授、および山谷研究室のみなさんに深く感謝致します。

最後に、日頃から熱心に討論、助言してくださいました愛知研究室のみなさんに深く感謝致します。