ecoWATTCH

Technological Feasibility Analysis
3/18/2025

Team ecoWattch

Sponsor
OP Ravi
Willow

Team Members
Collin Boyer
Avnish Kumar Sinha
Valentino Valero
Risa Walles

Mentor
Jeevana Swaroop Kalapala

Technological Feasibility Analysis

Table of Contents

Table of Contents **1**

1. Introduction **2**

2. Technological Challenges **4**

3. Technology Analysis **6**

3.1. Data Collection & Processing 6

3.2. Notification System 7

3.3. Gamification Engine 9

3.4. Mobile App Development 11

4. Technology Integration **14**

4.1. System Architecture Overview 14

4.2. System Diagram 15

5. Conclusion **16**

---

1. Introduction

In recent years, the rise in AI has undoubtedly been exponential, and with that comes a significantly higher priority in the expansion of data centers. As a result, global electricity consumption is at an all-time high with no sign of slowing down anytime soon. AI-integrated technologies alone account for roughly 4% of the globe's electricity use. Data centers, being the main consumers, are expected to consume 1000 terawatt hours on an annual basis by 2030. This increase in electricity consumption naturally results in a proportional increase in carbon emissions, making actions being taken now more important than ever if we'd like to preserve our favorite rock to live on. Our team recognizes this issue and has turned our attention to college campuses. Our campus, Northern Arizona University, alone spends roughly 15 million dollars annually in utilities, with more recent years averaging closer to 20 million dollars. As college students, this is very frustrating, not only because of the high emissions as a result, but also because our precious tuition money that we've poured into this university is being dumped straight into lights and computers that are simply sitting idle. These idle, unutilized electronics are the weak point that we aim to address with our project.

Today, most universities attempt to reduce electricity usage via campaigns, posted signs, and passive reminders for students. While these techniques may work to some extent, university higher-ups neglect the characteristics of college students, there's simply not enough incentive or direct reason for us to engage deeply in that way. To their credit, they've done a decent job bringing awareness to the problem, but ultimately it won't matter unless they can sustain engagement to continue bringing forth that change. *Willow*, our project sponsor, is directly involved in monitoring and managing electricity all the way down to individual units inside sectors of buildings and what they've realized is that traditional methods of energy conservation rely heavily on one's ability to interpret and understand the data they're looking at, it's simply not accessible in its current state. ecoWattch is our response to this. If we can effectively gamify electricity consumption in a way that appeals to college students from a college student perspective, we just may be able to significantly reduce electricity consumption and carbon emissions.

Our product focuses solely on dormitories, a high point of interest on campus. We know that college students love a sense of togetherness, teams, and friendly rivalry, so we've decided to lean heavily into this and put dorms against each other in a streak-based leaderboard game layout, each dorm duking it out to save more electricity via changing to LEDs, turning off lights at high usage times, etc, with the winners having tangible things to gain such as a blackout party (which saves even more electricity). By leveraging the things that make games fun, ecoWattch has the potential to make sustainability more engaging, bringing college students into the conversation, and keeping them there. Our sponsor OP Ravi has helped us greatly thus far, guiding our ideation processes, and as a result, we've landed on a few select features. Our app will feature a streak/points-based game structure that puts dorms directly against each other rather than students alone. With these points, dorms can find themselves competing on the leaderboard, with special "Power-Hours" (peak electricity usage times in a day) appearing to gain leverage on opposing dorms. Beyond the gamification aspects, one major feature we aim to achieve is to streamline the energy dashboard from Willow to our app, increasing accessibility by trimming out data and features that would be considered extraneous in the context of this app.

But of course, all of this is easier said than done. In this Technological Feasibility Analysis document, we will take a look into how we're going to address the technological challenges that may come with creating a product like this. We'll go through the technological challenges, the analysis of those challenges, and finally integration. But first, let's address the technological challenges.

---

2. Technological Challenges

The technical challenges we will face during this project should not be difficult to overcome. Many phone apps have been made before that use user log-ins, and many phone apps have been made that use databases. With that being said, there will be challenges that have to deal with user security and handling copious amounts of data regarding the users. These challenges are as follows:

1. **Security:** We need a secure way for our users to log in. This will include having two-step authentication and using a trusted database service to avoid data leaks. This is important not only for our users to trust our app and use it, but also important for the functionality of the app. It would be unfortunate if someone could get into our app's database and delete user data.

2. **Communication between apps and databases:** This will likely be the core issue that needs to be solved. The question of: "How is the app going to communicate with the necessary databases?" is an important one for the functionality of the app. Solving this problem will include finding a development kit that is robust enough to handle talking to a database quickly. The development kit should also be widely used, so there is support for it and external packages written for it. The system that is set up for this also needs to be scalable to allow many users to sign up for the app with no issues.

3. **Speed of user interface and data retrieval:** No one wants to use an app that is unresponsive and slow, or takes a long time to load everything that is needed. In order for our app to be used by its users, it needs to be responsive, and load views and activities in seconds. This requires the development kit we find to be lean and not have much overhead. The database also needs to be accessible quickly. Solving this problem will include finding a development kit that allows for multi-threading, as well as creating an app that keeps apps lean by not needing copious amounts of files to make it work. This will also include finding a database that can be queried quickly

4. **Synchronization between users:** Our app will consist of electricity data that is taken over the course of weeks to look for trends and patterns. For example, the app might tell a user: "Your dorm has used 1000 kilowatt-hours of electricity this week". This data needs to be synchronized across all the apps, so there needs to be a standard time when all the apps update their data from the Willow database. This synchronization is needed because for the game aspect of the app, it is needed that users can see how they are competing against other dorms or buildings.

5. **New technologies**: There are lots of new technologies that we will be using during the development of this app. Our team has never worked with many of the apis that we will be using, such as the willow API. We will need to answer questions such as: "how will we make the phone app talk to the willow database" or "how will we display the information that we get from the database in a way that our sponsor is happy with". These challenges are fundamental to our development because we are required to use these new technologies to make the app that our sponsor wants.

6. **UI:** An app needs a user friendly interface to be successful and to be used. Our team will need to design a UI that is not only intuitive, but also at least somewhat fun to use. This is vital for our project because it will be used by students, and everyone needs some engagement from their apps to keep them coming back. No one wants to come back to a boring or hard to use application.

7. **Cheap:** We don't want to spend any money that we don't have to. This means finding preferably cheap if not free alternatives for whatever technologies we need to use.

---

3. Technology Analysis

To ensure the feasibility and effectiveness of our solution, we analyze the critical technological decisions and their implications. The main areas of focus include data processing, notification delivery, gamification, and integration with existing systems.

3.1.1 Data Collection & Processing

This issue stems from the idea of needing to store the data given to us by users. This problem is solved by selecting a reliable and efficient database system. Our project demands a database that can handle frequently updated data such as electricity consumption readings per building, user profile data, and accumulated points. The chosen database must support rapid reads and writes, and allow for smooth interaction with the mobile app. Given these requirements, we have selected **AWS RDS MySQL**, a managed relational database service that offers scalability, security, and ease of integration.

3.1.2. Desired Characteristics

The characteristics that the database needs are as follows:

- **Time-sensitive data retrieval**: The data must be accessible at any time with minimal delay.
- **Scalable**: The system should allow for easy expansion as new users sign up and more data is collected.
- **Low overhead**: Efficiency is key; we need minimal latency during queries and updates.
- **Security**: Data must be encrypted and access-controlled to protect user privacy and system integrity.

3.1.3. Alternatives/Analysis

To evaluate the best database solution for our real-time data storage and gamification needs, we reviewed the following:

- **Neo4j (Graph Database)**: Great for relational data, but unnecessary complexity for our time-based and transactional needs.

- **PostgreSQL**: Strong candidate for relational data, but our team opted for more seamless integration and management with AWS.
- **AWS (Relational Database)**: Offers tight integration with other AWS services like Lambda, API Gateway, and CloudWatch, making it an ideal choice for scalable, secure, and cloud-native application development.

Key Findings:

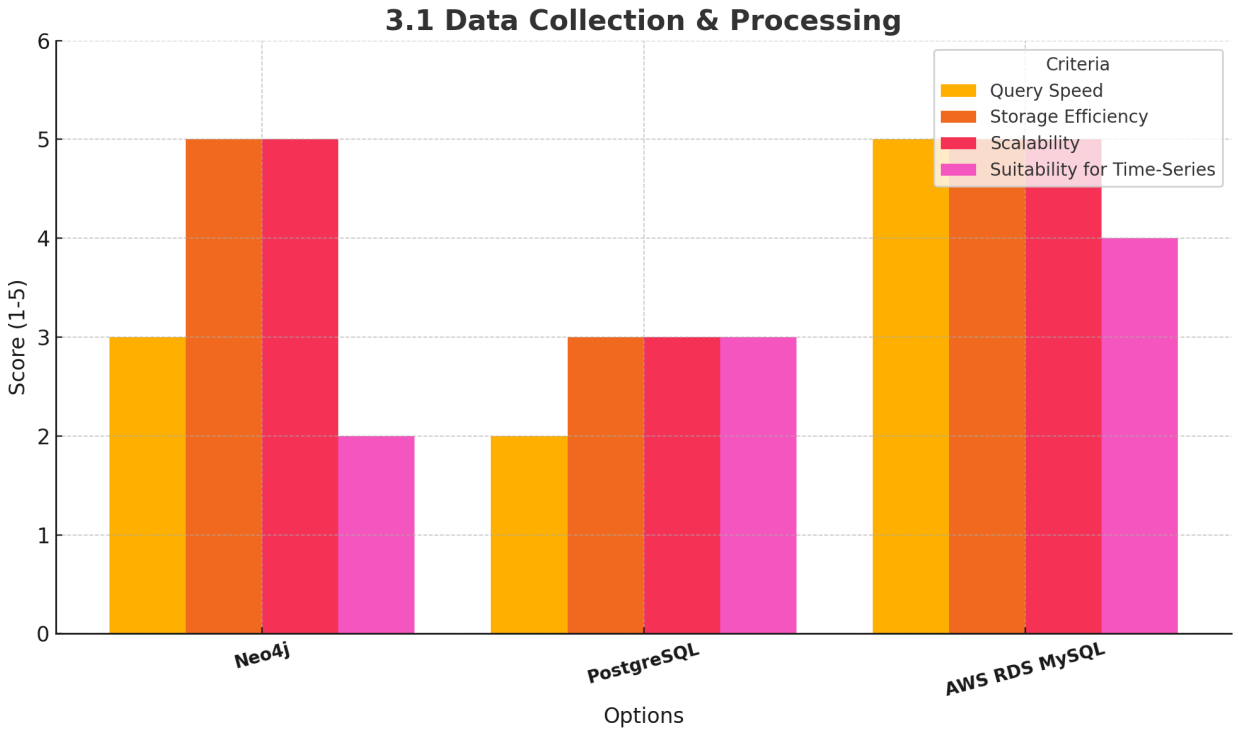
- **Neo4j** is optimized for complex relationships, but it adds overhead and complexity we do not need for our use case.

- **PostgreSQL** is powerful and flexible, but it requires more manual setup and does not provide the seamless scalability of a managed solution.

- **AWS RDS MySQL** offers a strong balance of performance, manageability, scalability, and cost-effectiveness for our specific needs.

3.1.4. Chosen Approach

We selected **AWS RDS MySQL** for its ability to efficiently store and retrieve structured data while offering managed database services, automated backups, and high availability. It provides SQL querying capabilities needed for user interactions, gamification logic, and real-time leaderboard tracking. With the scalability of AWS infrastructure, we can support user growth while ensuring low-latency responses for app users.



3.1.5. Proving Feasibility

An initial prototype Android application will be developed and tested for core functionalities. This will involve building a minimal viable product (MVP) with essential features such as user authentication, energy data visualization, notifications, and gamification elements. The prototype will be tested across different Android devices to ensure compatibility, performance, and user experience. Feedback from early testers will be collected to refine the application before full-scale deployment.

3.2. Notification System

3.2.1. Introduction

This issue is the idea that we need to be able to send the users notifications. These notifications are somewhat complex, as to which device gets sent a notification depends on the database data. This will be a crucial part of our application because we need to be able to send users notifications about electricity consumption, or if they have won an in-game contest, or perhaps a weekly update on their electricity consumption. The functions that this system will need to handle include querying the database, looking up information about specific users, and sending notifications to those users based on what it finds in the database.

3.2.2. Desired Characteristics

The characteristics that the database needs are as follows:

- **Time-sensitive delivery:** We need the notifications to be able to be sent quickly, so if the user gains a certain number of points, they can be notified right away.
- **Scalable:** We need the delivery system to be scalable to the point where we can easily add users. We don't need it to be scalable to a whole state, but we need some flexibility in the number of users because a user can sign up at any time.
- **Low overhead:** We need the system to be somewhat lean when queried and used. A phone is a powerful machine, but we want this app to be as lean and fast as possible.
- **Free:** We want this system to be free, as there are free options available.

3.2.3. Alternatives/Analysis

Two notification systems were evaluated:

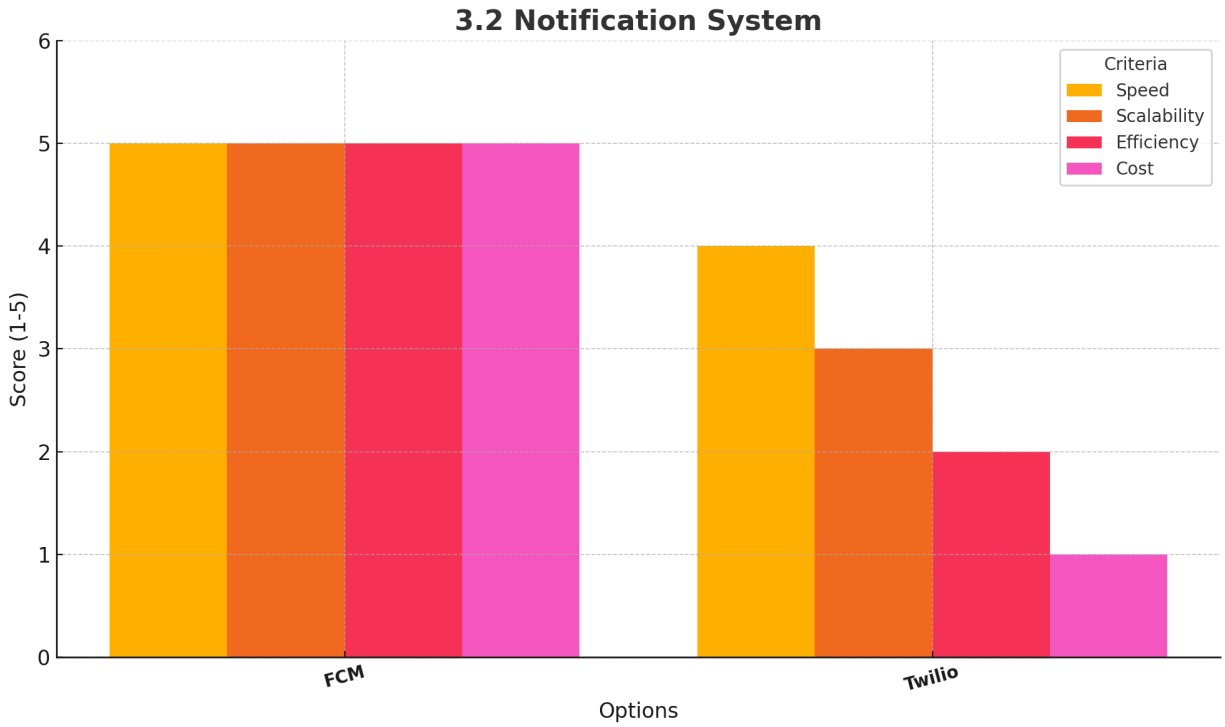
- **Firebase Cloud Messaging (FCM)**: Real-time, cross-platform, cost-effective, and integrates easily with Android.
- **Twilio SMS Alerts**: Reliable for text-based messaging, but incurs costs as usage scales.

Key findings:

- **FCM** provides the best balance of scalability, cost, and performance for mobile notifications.
- **Twilio,** while reliable, is not cost-effective for high-frequency notification requirements.

3.2.4. Chosen Approach

FCM is chosen due to its instant delivery, scalability, and cost efficiency. It enables real-time push notifications to users, ensuring timely alerts on energy usage. Additionally, it supports cross-platform messaging, making it a robust choice for engaging users effectively without incurring significant costs.



3.2.5. Proving Feasibility

A test application will be deployed to verify notification efficiency under various conditions. This includes assessing message delivery times, success rates, and reliability during peak loads. We will simulate different energy scenarios to evaluate how FCM handles real-time notifications and ensure that critical alerts are delivered to users without delays. Additionally, we will monitor user engagement levels to refine notification strategies.

3.3. Gamification Engine

3.3.1. Introduction

This part of the project is about the game and how exactly the game will be implemented. How exactly we make this game is important because the game is the whole incentive for students to reduce their electricity consumption. The game aspect is something that the client really wants to connect with college students. The functions this will include will be features such as a point system, having customization options to spend the points a user collects, and how exactly points are collected.

3.3.2. Desired Characteristics

The characteristics that the database needs are as follows:

- **Time-sensitive delivery:** We need this system to be very fast in delivering the points. If a user earns points, they should be able to see the points they have earned right away.
- **Scalable:** We need the point system to be scalable to the point where we can easily add users. We don't need it to be scalable to the point of a whole state, but we need some flexibility in the number of users due to the fact that a user can sign up at any time.
- **Low overhead:** We need the system to be somewhat lean when it comes to being used. A phone is a powerful machine, but we want this app to be as lean and as fast as possible.
- **Free:** We want this system to be free, as there are free options available.

3.3.3. Alternatives/Analysis

Evaluated three options for implementing gamification:

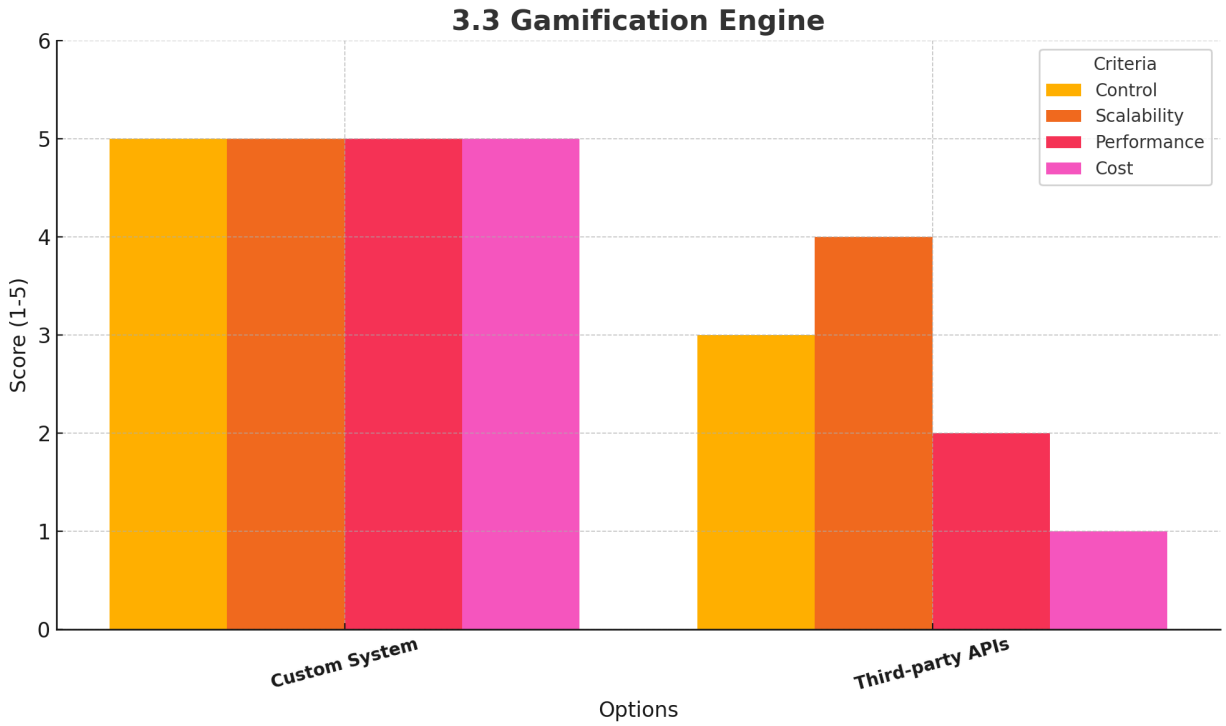
- **Custom point-based system**: Lightweight and simple to implement, offering control over logic and customization.
- **Third-party APIs (e.g., Badgeville, Bunchball)**: Feature-rich but expensive and unnecessarily complex.
- **Gamification libraries**: Some offer templates, but can add bloat or limit flexibility.

Key findings:

- Third-party APIs are costly and often over-engineered for our needs.
- A custom point system is easier to develop, maintain, and tailor to our app's structure.

3.3.4. Chosen Approach

A basic point-based system will be used for the MVP to encourage user engagement and drive behavior change. This system is easy to implement, tracks energy-saving activities, and rewards users based on their contributions. It fosters gamification elements that align with sustainability goals by providing incentives for reducing energy consumption. This system does not need to be complicated, so we may only use external APIs for the data the system creates.



3.3.5. Proving Feasibility

A basic point system prototype will be implemented and tested with user engagement simulations. This will include creating a small-scale leaderboard where test users can earn and redeem points for simulated energy-saving activities. We will analyze how different incentive structures influence engagement and optimize the system for maximum effectiveness. Additionally, we will conduct usability tests to ensure a seamless experience.

3.4. Mobile App Development

3.4.1. Introduction

What system or development kit we use to develop this app is important to consider. The development kit should be robust enough to be able to implement the features we want. The role the development kit will play in our project will mostly be for us as developers. A development kit is needed that would not only support the complex features we intend to implement but also provide a seamless experience for the development team. A development kit is needed that would allow us to rapidly prototype and build a stable, high-performance app.

3.4.2. Desired Characteristics

The characteristics that the database needs are as follows:

- **Flexibility of objects:** We need the development kit to offer easily reproducible objects that we can copy. This will speed up development by allowing us to copy work that we have done previously.
- **Low overhead:** We need the system to be somewhat lean when it comes to being used. A phone is a powerful machine, but we want this app to be as lean and as fast as possible.
- **Free:** We want this system to be free, as there are free options available.
- **Commonly used:** We need the development kit and tools to be very widely used, as we will need extra documentation and APIs to help us in the future.

3.4.3. Alternatives/Analysis

Evaluated three approaches for mobile app development:

- **Standalone Android App (Android Studio):** Provides full control over features but requires independent development.
- **Hybrid App (React Native, Flutter):** Faster development, but performance trade-offs.
- **Progressive Web App (PWA):** Mobile-friendly but with limited native functionality.

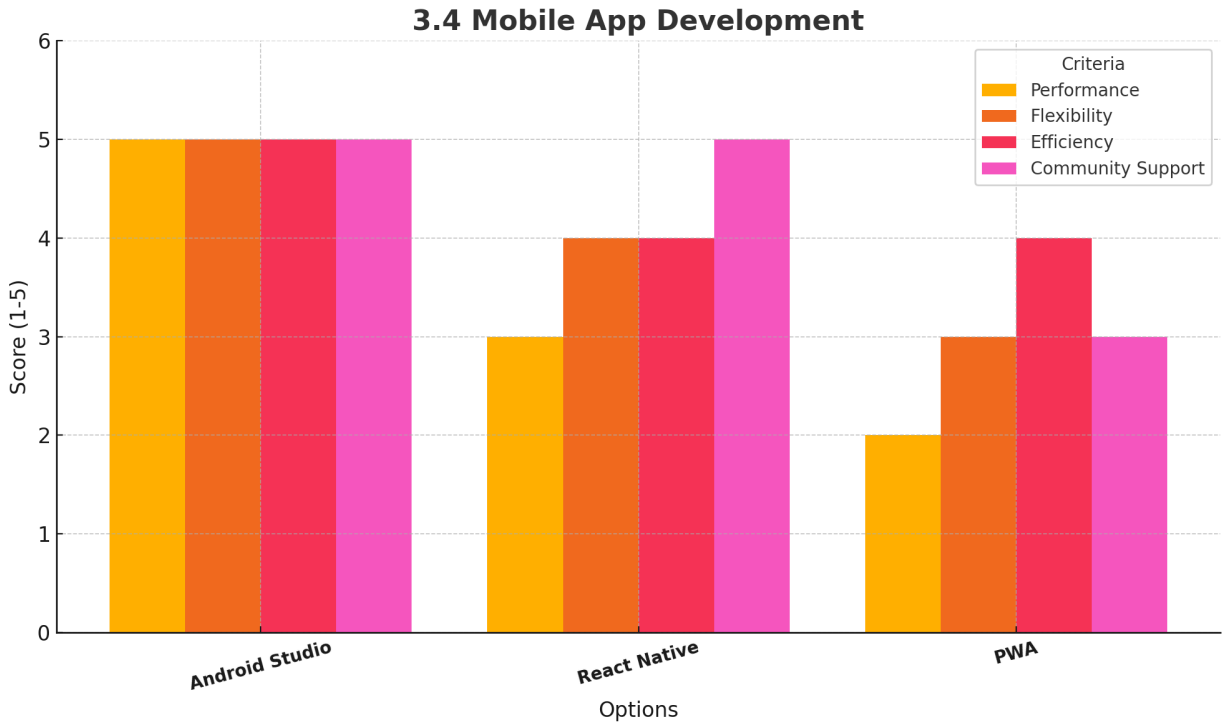
Key findings:

- Android Studio provides native performance and deep integration with Firebase and device APIs.

- Cross-platform tools add development speed but may compromise feature depth.

3.4.4. Chosen Approach

We aim to develop a dedicated Android application that provides full control over features, user experience, and performance. By opting for a standalone app, we ensure optimized resource management, better security, and customization tailored to sustainability-focused functionalities. This approach allows for a seamless, scalable, and independent system without external dependencies.



3.4.5. Proving Feasibility

An initial prototype Android application will be developed and tested for core functionalities. This will involve building a minimal viable product (MVP) with essential features such as user authentication, energy data visualization, notifications, and gamification elements. The prototype will be tested across different Android devices to ensure compatibility, performance, and user experience. Feedback from early testers will be collected to refine the application before full-scale deployment.

---

4. Technology Integration

4.1. System Architecture Overview

To effectively integrate the components, we propose a modular architecture where different functionalities interact through defined interfaces. The system will be structured as follows:

- **User Authentication & Profile Management**
  Users log in to the app and are assigned to specific buildings. Authentication will be handled independently within the application.

- **Data Collection & Processing**
  Willow's API will collect real-time energy consumption, pricing, and emission data from building meters and external sources. This data will be processed and stored in an **AWS RDS MySQL** relational database. The database schema will be designed to efficiently handle time-stamped readings, user profiles, and game point transactions. MySQL's transactional support and query flexibility make it suitable for analyzing trends, retrieving historical data, and enabling leaderboard tracking.

- **Notification System**
  Users receive alerts when a building's energy consumption or emissions exceed predefined thresholds. Peak load notifications will be sent separately with recommendations for load reduction.

- **Gamification Engine**
  Users earn points based on energy savings during specific "power hour" periods. Points can be redeemed for virtual or physical rewards provided by the sustainability department. All user actions and point transactions are recorded in AWS RDS MySQL.

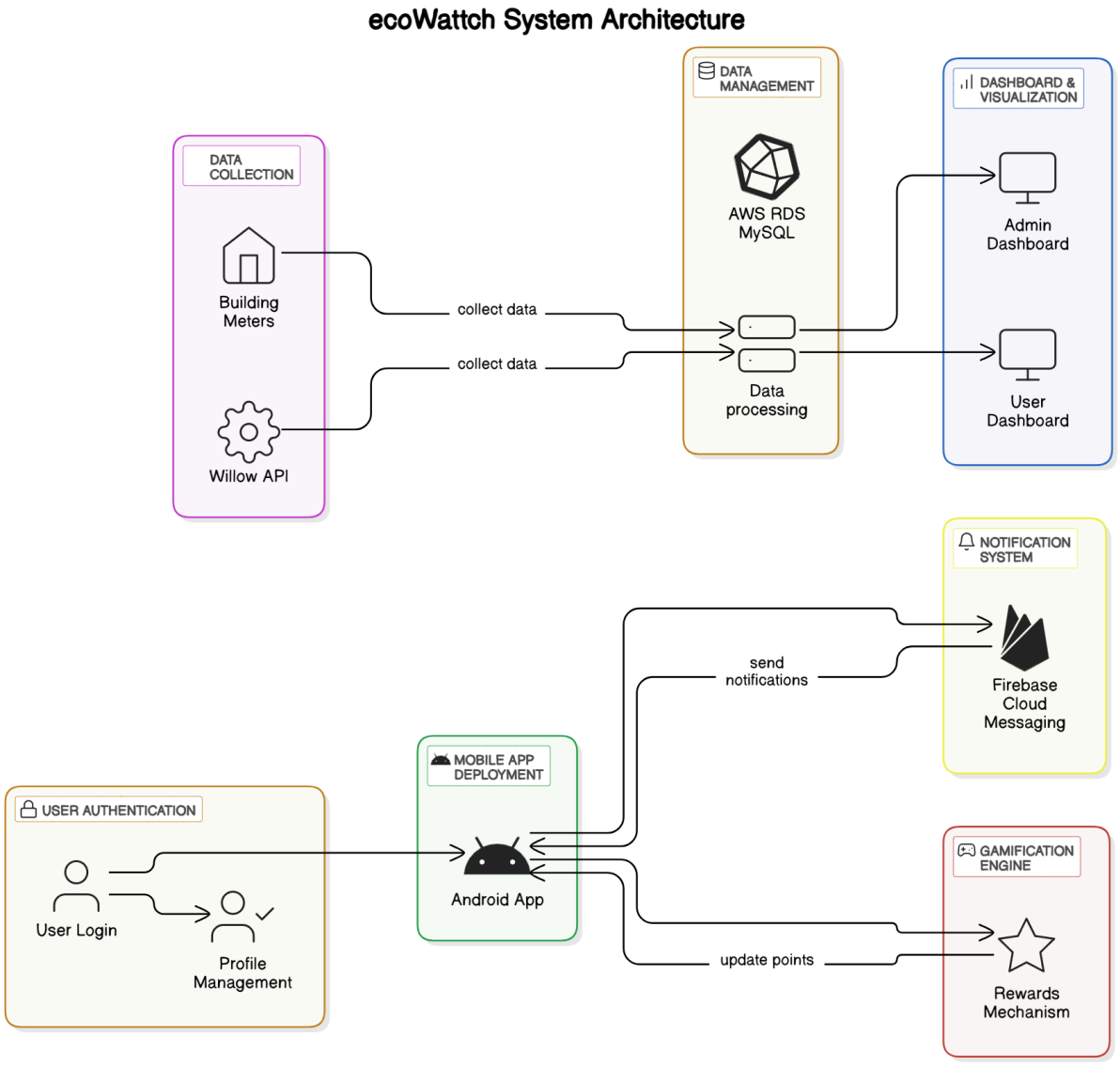
- **Dashboard & Visualization**
  Users and administrators can view energy consumption trends and comparisons between buildings. Data will be visualized using bar graphs and other visual elements for enhanced awareness. Backend APIs will fetch data from AWS RDS MySQL to power the mobile app's visualizations.

- **Mobile App Deployment**
  The application will be developed as a standalone Android application. The app will be distributed independently and will not integrate with any external systems other than Willow's API and AWS backend services.

4.2. System Diagram

A system diagram visually represents how these components interact, illustrating data flows, API connections, and user interactions. The final document will include a diagram to show how the elements fit together seamlessly.

By integrating these components effectively, we ensure the system is scalable, efficient, and ready for deployment to drive sustainability awareness and behavioral change.



5. Conclusion

Our project aims to tackle the issues of overconsumption of electricity on NAU campus. Energy usage has gone up in recent years, especially with the introduction of generative AI like ChatGPT. This increase in energy usage leads to increases in carbon emissions, which is ultimately bad for the environment. NAU students use a lot of electricity, but many attempts to get students to reduce their energy usage in the past have not been incredibly effective. By creating a mobile app that gamifies energy conservation and provides incentives to watch electricity use, we hope to decrease the overall energy usage of students on campus, leading to a smaller effect on the environment.

In our feasibility analysis, we examined four main areas of focus and determined what systems would work the best for our purposes. For data storage, we chose the time-series database InfluxDB for its optimized storage and retrieval of time-based data. For the notification system, we opted for using Firebase Cloud Messaging (FCM); as it is fast, easy to integrate, and the more cost-effective option. To actually implement the game, we examined a few APIs but ultimately decided to use a simple point-based system; this option is scalable and easy to implement, but also easy for users to understand. Finally, we chose to develop a standalone Android app for mobile deployment. This will be difficult to develop, but provides us with the most control over features and functionality.

As a group, we feel confident in our ability to progress with this project using the technology we have outlined. Our next steps moving forward will be to work on the Requirements Specifications Document and technical demos. The tech demos will be very useful - if we find that any of the technology we've selected does not work as well as expected, this will give us an opportunity to switch to one of our backup options.
