Faculty of Natural and Mathematical Sciences
Department of Informatics

King's College London
Strand Campus, London,
United Kingdom

7CCSMPRJ

Individual Project Submission 2023/24

Name:	Ashmit Khadka
Student Number:	23025270
Degree Programme:	MSc Individual Project
Project Title:	Software Platform for Promoting Sustainable Practices
Supervisor:	Dr Leonardo Magela Cunha
Word Count:	14986

RELEASE OF PROJECT

Following the submission of your project, the Department would like to make it publicly available via the library electronic resources. You will retain copyright of the project.

Check the appropriate box below

☐ I **agree** to the release of my project

☐ I **do not** agree to the release of my project

Signature: Signature **Date:** August 13, 2024

Department of Informatics
King's College London
United Kingdom

7CCSMPRJ Individual Project

Software Platform for Promoting Sustainable Practices

Name: **Ashmit Khadka**
Student Number: 23025270
Course: MSc Individual Project

Supervisor: Dr Leonardo Magela Cunha

This dissertation is submitted for the degree of MSc in MSc Individual Project.

Acknowledgements

I wish to express my gratitude to Dr Leonardo Magela Cunha for his support and guidance throughout this project. Dr Cunha's invaluable feedback during our progress meetings has been instrumental in shaping the direction of the project, ensuring that the deliverables remain focused, relevant and add value to the realm of sustainable development and software engineering.

I would also like to thank my temporary supervisor, Dr Eamonn Postlethwaite, for his insightful contributions to the project's structure and for his availability to address my queries in short notice.

Additionally, I would like to thank King's College London for providing access to a range of literature and facilities, which have been essential for the successful completion of this project.

Abstract

This report focuses on developing a software platform aimed at enhancing engagement in sustainable development practices, particularly within the higher education settings. The primary objectives include motivating users to adopt and maintain sustainable behaviors through a user-centered design approach and utilising emerging technologies to create a robust and scalable software solution. The application leverages the MERN stack in conjunction with the VIPER architecture. It incorporates gamification, AI integration, and interactive modules to translate complex sustainability concepts into actionable and rewarding user experiences (UX). Challenges addressed include system scalability, AI integration, performance, and UX. The report details the methodologies, technologies, and design principles employed to overcome the challenges inherent in developing such an application, aiming to foster a community that is committed to sustainable development through innovative and engaging digital solutions.

Contents

List of Figures

List of Tables

1 Introduction

Sustainable development is increasingly recognised as a crucial global priority, driven by the pressing need to address environmental degradation, climate change, and the scarcity of resources. This recognition has evolved over decades, with the concept of sustainable development becoming central to global policy frameworks and initiatives [1]

Despite widespread awareness of these challenges, many individuals and organisations struggle to convert this understanding into practical steps that promote sustainability in their daily lives. Traditional methods of sustainability education often fall short, lacking the engagement and motivation necessary to inspire meaningful action [2]. This project seeks to address this gap by developing an innovative software solution that presents users with personalised sustainable development challenges, encouraging them to adopt and sustain environmentally friendly practices.

This report details the development of a software application aimed at fostering sustainable behaviour through interactive and gamified challenges. By harnessing modern technologies such as AI, user-centred design and cutting edge software development technologies. The software aspires to make sustainability more accessible, engaging, and rewarding for a diverse audience. The following sections of this report will delve into the project's motivation, the problem it seeks to solve, and its relevance to the fields of sustainable development and software engineering.

1.1 Motivation

Sustainable development represents a significant global challenge demanding immediate and ongoing action. Environmental degradation, climate change, and resource depletion pose existential threats meaning it is important to engage individuals, communities, and organisations in adopting more sustainable practices. Traditional methods of promoting sustainability, however, often fail in motivate long-term behavioural change. Despite increasing awareness of sustainability issues, a considerable gap still persists between knowledge and action. Agbedahin [2] argues the need for more dynamic and engaging methods to promote sustained behavioural change. Bridging the gap from knowledge to action requires innovative approaches that not only educates but also actively engages people in a manner that produces consistent and meaningful participation in sustainable development.

A promising approach to enhancing engagement is the concept of competition, a core essence of gamification. Competition has potential to create dynamic environments where individuals are driven by their desire to outperform others which is a motivation that can be grasped to encourage sustainable behaviour. By incorporating competitive component, people may be more committed to engage with sustainability challenges, driven by prospect of achieving higher rankings, earning rewards or gaining recognition among peers. While the effectiveness of competition in this context remains theoretical, its potential to enhance engagement and instil a sense of accomplishment makes it an intriguing area for exploration. A software solution provides an ideal platform for implementing such competitive elements. It can offer scalability, personalisation, and the

ability to track and compare progress in real-time. Such software could be a powerful tool to bridge the gap between awareness and action, making sustainability efforts more engaging, accessible, and impactful.

1.2 Problem

Sustainable development is a holistic approach which can be traditionally described as meeting the needs of the present without compromising the ability for future generations to meet their own needs [3]. The concept, though widely accepted, is complex and has evolved over decades, encompassing various factors that include environmental, social, and economic considerations [2] . In 2015, the United Nations introduced the 17 Sustainable Development Goals (SDGs) as a universal call to action to end poverty, protect the planet, and ensure peace and prosperity for all by 2030. Here are a summary of each of the goals:

1. **No Poverty**: End poverty in all its forms everywhere.
2. **Zero Hunger**: Achieve food security, improve nutrition, and promote sustainable agriculture.
3. **Good Health and Well-being**: Ensure healthy lives and promote well-being for all at all ages.
4. **Quality Education**: Ensure inclusive and equitable quality education and promote lifelong learning opportunities for all.
5. **Gender Equality**: Achieve gender equality and empower all women and girls.
6. **Clean Water and Sanitation**: Ensure availability and sustainable management of water and sanitation for all.
7. **Affordable and Clean Energy**: Ensure access to affordable, reliable, sustainable, and modern energy for all.
8. **Decent Work and Economic Growth**: Promote sustained, inclusive, and sustainable economic growth, full and productive employment, and decent work for all.
9. **Industry, Innovation, and Infrastructure**: Build resilient infrastructure, promote inclusive and sustainable industrialisation, and foster innovation.
10. **Reduced Inequality**: Reduce inequality within and among countries.
11. **Sustainable Cities and Communities**: Make cities and human settlements inclusive, safe, resilient, and sustainable.
12. **Responsible Consumption and Production**: Ensure sustainable consumption and production patterns.

13. **Climate Action**: Take urgent action to combat climate change and its impacts.
14. **Life Below Water**: Conserve and sustainably use the oceans, seas, and marine resources for sustainable development.
15. **Life on Land**: Protect, restore, and promote sustainable use of terrestrial ecosystems, manage forests sustainably, combat desertification, halt and reverse land degradation, and halt biodiversity loss.
16. **Peace, Justice, and Strong Institutions**: Promote peaceful and inclusive societies, provide access to justice for all, and build effective, accountable, and inclusive institutions at all levels.
17. **Partnerships for the Goals**: Strengthen the means of implementation and revitalise the global partnership for sustainable development.

Despite growing awareness of sustainable development, there remains a gap in effectively engaging individuals and organisations in meaningful, long-term sustainable practices. Traditional educational methods and tools often fail to inspire action, leaving people feeling overwhelmed and uninspired. This is particularly noted in higher education institutions, where the integration of sustainability into the curriculum and daily operations is often hindered by lack of resources, institutional barriers and ineffectively engaging content.

The problem is also compounded by the fact that sustainability is a multifaceted concept, requiring constant engagement and clear understanding of implications. Many struggle with the abstract nature of sustainability and the perceived lack of immediate benefits from adopting sustainable practices. As a result, there is a need for innovative solutions that can effectively translate sustainability concepts into actionable, engaging, and rewarding experiences.

1.3 Aims

The primary aims of this project are as follows:

- **Tackling Sustainable Development:** Increase engagement and participation in sustainable development practices by motivating users to adopt and maintain sustainable behaviors.
- **User-Centered Design:** Address a highly interactive software engineering challenge through a user-centered approach, ensuring that the software meets the needs and expectations of its users.
- **Utilising Emerging Technologies:** Leverage cutting-edge technologies such as No-SQL databases, Single Page Applications (SPAs), and advanced API query and manipulation languages like GraphQL to develop a robust and scalable software solution.

2 Literature Review

2.1 Perception of Sustainable Development

The perception of sustainable development in higher education can be understood by exploring the mental models of today's students. A word cloud is shown illustrating the 50 most frequent words associated with sustainability [4]. A particularly prominent term is "environment" showing the primary association youth have with sustainability. However, the term "sustainability" by definition is broad and encompasses a range of topics. This suggests that youth perceive sustainability through a relatively narrow lens. It becomes more evident when the survey delves into the specific terms they are familiar with.

Moreover, there is a notable disparity in perceived importance among various topics. For instance, climate change is ranked significantly higher in priority compared to topics such as good health, decent jobs, and economic growth. The UN's sustainability goals aim for each topic to be valued equally [4].

The report shows that terms like "ethical investment", "circular economy" and "rewilding" are not well understood by students. This might explain the gap in perceived importance and prominence. It might be of importance to introduce these concepts in a more digestible way and create a more comprehensive understanding of sustainability.

The majority of students also believe that current efforts towards sustainable development are ineffective. A significant proportion also believe that the responsibility for achieving sustainability lies with everyone. Accountability is evident among students, as most recognise that each individual plays a crucial role in fostering a more sustainable future [4].

This might explain why we see a significant level of "eco-anxiety" among young people. In the sentiment analysis 87% of answers fall into "negative" or "slightly" negative when asked about the future. The perception that current efforts are ineffective combined with accountability for actions might play a role in the anxiety levels we see. It could be important to consider how we can create a positive and encouraging environment in the software solution.

2.2 Barriers to Sustainable Development in Higher Education

Universities play an important role in transforming societies, a reflection of the nature of universities and their mission. Research involving 269 Higher Education Institutions (HEIs) identified 25 key obstacles hindering sustainable development. The study revealed that the lack of support from management was the significant barrier in implementing sustainable development (SD) [5]. This can show a bottom-up motivation for SD within HEIs, where individual stakeholders are driving the cause. However, administrative constraints hinders effective integration of a comprehensive SD program into the curriculum [6]. Limited authority of teachers further contributes to the lack of managerial support in executing ideas. Other barriers were similar in significance, indicating that universities encounter a combination of challenges.

Two particularly noteworthy barriers were the lack of appropriate technology and insufficient awareness or concern, ranking second and third respectively [5]. The findings indicate an infrastructure issue, where current technology falls short in supporting SD activities. Innovative approaches that leverage existing infrastructure could play a critical role in making sustainability more accessible in higher education [7]. Additionally, the lack of awareness and concern aligns with the general mental model described earlier. For students to engage in SD, it must be ingrained in the social culture. It cannot be assumed that teachers and administrators possess the necessary values and virtues to establish this culture [8]. Therefore, education initiatives must be in place to guide students in the right direction.

Another significant factor was the absence of an environmental committee [5]. Without a central committee to implement structured practices and policies, stakeholders are left to develop action plans independently. Streamlining the communication process with stakeholders within the organisation is a critical consideration.

This study does have limitations. For example, the survey was conducted with a relatively small sample of professionals working in SD within higher education. A separate study examined the use of gamification in games and apps to promote sustainable practices. For instance, the board game "Factory Heroes" showed that graduate students improved their leadership in sustainability related to manufacturing [9]. Another game, "Keep Cool" was found to increase youth's sense of personal responsibility for sustainability [10]. Overall, the study concluded that gamification, particularly through board games, can be an effective method for educating young people about sustainability.

Moreover, apps that incorporated gamification elements, such as point systems, were generally rated more favorably than those that merely provided information [11]. Techniques like nudging, combined with gamification, have proven to be successful in achieving long-term engagement in the context of sustainability [12].

Psychological theories also provide insight into why gamification is effective. For example, Gifford's theory of the 29 "dragons" to pro-environmental behaviors identifies psychological barriers that reduce sustainable practices [13]. Gamification can address these barriers; for instance, instructional gamification engages users and informs them of sustainable behaviors they can adopt, thereby reducing the ignorance barrier.

2.3 Promoting Sustainable Development Practices

The SHIFT framework is a model for conceptualising and promoting pro-sustainable behaviors. SHIFT emphasises five key tactics to influence consumer behavior towards sustainability: Social Influence, Habit Formation, The Individual Self, Feelings and Cognition, and Tangibility. Although initially designed for marketing, the values and concepts within the SHIFT framework are highly relevant to software design, particularly in creating impactful, user-centered applications.

The framework underscores the importance of social influence, particularly in guiding users based on perceived social norms [14]. Rewarding users who adhere to sustainable norms can be a key strategy for engaging them in sustainable practices. The framework also discusses "Social Desirability," where consumers choose sustainable options to create

a positive impression on others [15]. Additionally, it mentions "social identities," formed through group membership, making individuals more likely to engage in activities which are common within their group [16].

Many unsustainable behaviors are shaped by habits formed over time. The SHIFT framework suggests using strategies like prompts, incentives, and feedback to cultivate pro-sustainable habits where simplicity is a crucial element [17].

Self-impact refers to how users perceive themselves and how they wish to be perceived by others, which influences their behavior [18]. For instance, the concept of "self-consistency" suggests that people want to see themselves positively (e.g., as environmentally concerned) [19]. Another relevant concept is "self-efficacy" where users' perceived potential impact on sustainability affects their overall engagement [20].

Emotions also play a significant role in the SHIFT framework. Positive emotions such as joy, pride, and hope should be reinforced to encourage sustainable practices [21]. While intense negative emotions should be avoided, mild feelings of fear, guilt, and sadness can motivate sustainable attitudes, provided they do not overpower positive emotions [22].

Finally, Tangibility is perhaps the most intriguing concept. Sustainability, by its nature, is future-oriented, which can create a sense of distance and abstraction that makes it difficult for users to relate to [23]. Therefore, it is crucial to communicate the local and immediate impacts of sustainability, along with concrete steps that users can take. Encouraging a future-focused mindset can help overcome feelings of distance and uncertainty [24].

2.4 Insights

From on the literature review, here are some insights which might be relevant in the context of creating a software engineering solution. These can serve as a reference point when ideating user stories.

1. Need for Clear Goals and Progression: Users need clear progression paths and achievable goals to stay motivated in sustainability initiatives

2. Preference for Social Interaction and Competition: Users are more likely to engage in sustainable behaviors when they can compare their actions with others and compete in a friendly manner.

3. Desire for Immediate Feedback: Immediate feedback on their actions helps users understand the impact of their behaviors and stay motivated to continue.

4. Feeling of Responsibility: Users, especially youth, feel a heightened sense of personal responsibility towards sustainability when they are involved in interactive and participatory methods.

5. Barriers in University Settings: Users in universities face institutional barriers such as limited resources and resistance to change, which hinder the implementation of sustainable practices.

6. Need for Integration into Curriculum: Users express the need for sustainability to be integrated into the academic curriculum to foster a deeper understanding and commitment.

7. Lack of Awareness and Knowledge: Users often lack awareness and knowledge about sustainable behaviors, highlighting the need for educational interventions (Obstacles to Sustainable Development at Universities.

8. Engagement through Gamified Apps: Users are more engaged with sustainability topics through gamified apps that incorporate fun and interactive elements.

9. Need for Practical Solutions: Users look for practical solutions and tools that can help them implement sustainable behaviors in their daily lives.

10. Influence of Social Norms: Users' behaviors are significantly influenced by social norms and the actions of their peers.

11. Motivation through Rewards: Users are motivated by rewards and recognition for their sustainable actions, which can be effectively implemented through gamification.

12. Concerns about Privacy: Users have concerns about privacy and data security when using apps that track their behaviors.

13. Need for Continuous Engagement: Continuous engagement and regular updates are necessary to keep users motivated and prevent drop-off.

14. Desire for Customisation: Users appreciate apps that offer customisation and allow them to set their own sustainability goals.

15. Emotional Connection to Sustainability: Users form a stronger commitment to sustainability when they feel an emotional connection to the cause.

16. Role of Education in Behavior Change: Educational interventions that provide information about the impact of behaviors can effectively change users' actions.

17. Importance of Visualising Impact: Users benefit from visualising the impact of their actions on the environment, which can be facilitated through gamified apps.

3 Objectives, Specification and Design

3.1 Personas

Personas are research based fictional characters that represent various potential users for the app. They ensure the diverse needs and motivations of our users guide the design and development process. There are many ways of creating personas, our aim is to use personas to drive requirement analysis. Therefor, we empathise to visualise pain points, needs, and feelings asking ourselves what does this user need? and why? With these insights, we can tailor the features, user interface, and overall experience to better meet their needs [25]. By prioritising focus around features that resonate with the target audience, we aim to reach higher user engagement and satisfaction, key issues raised in the problem statement. Here are the persona's of Emma, Adnan and Sarah based on qualitative insights obtained from the literature review. Each persona offers a unique perspective on the app, providing a broader understanding of user needs. For instance, Emma is a busy undergraduate student, while Adnan is working to bring change to his organization.

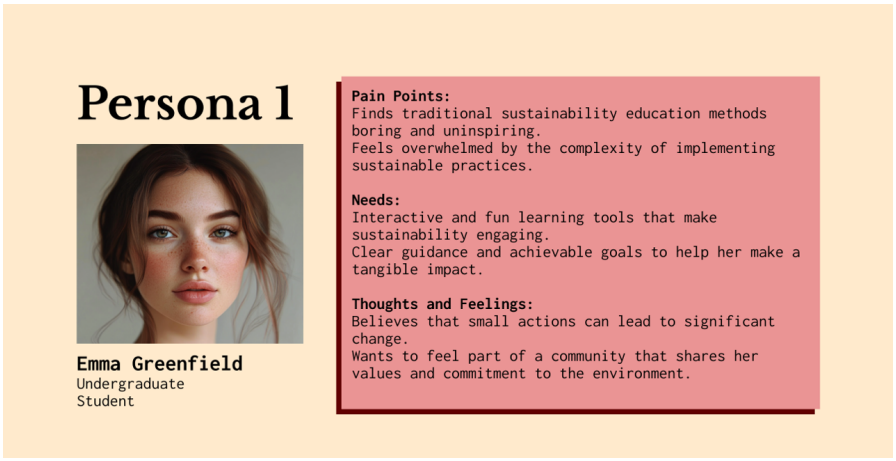

Figure 1: Persona 1 - Emma Greenfield, undergraduate student

While this project aims to meet the needs of as many users as possible, required to make an impact towards SD, the target audience are students in higher education. As such Emma, will be the main focus.

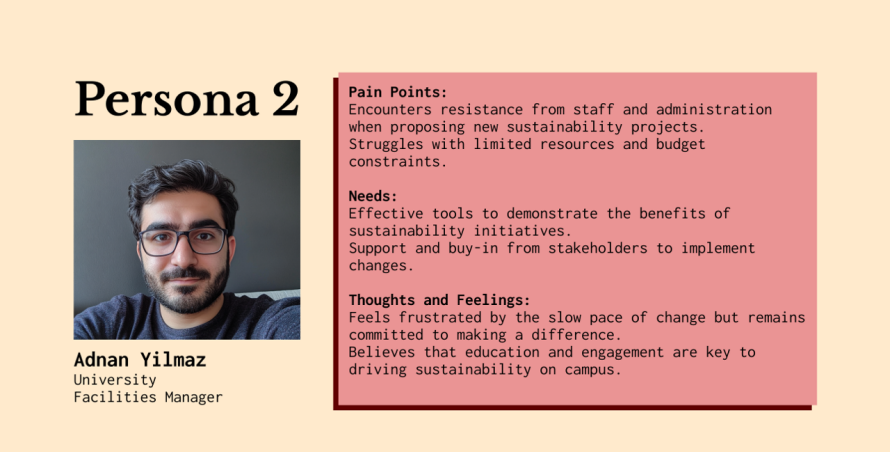

Figure 2: Persona 2 - Adnan Yilmaz , university facilities manager

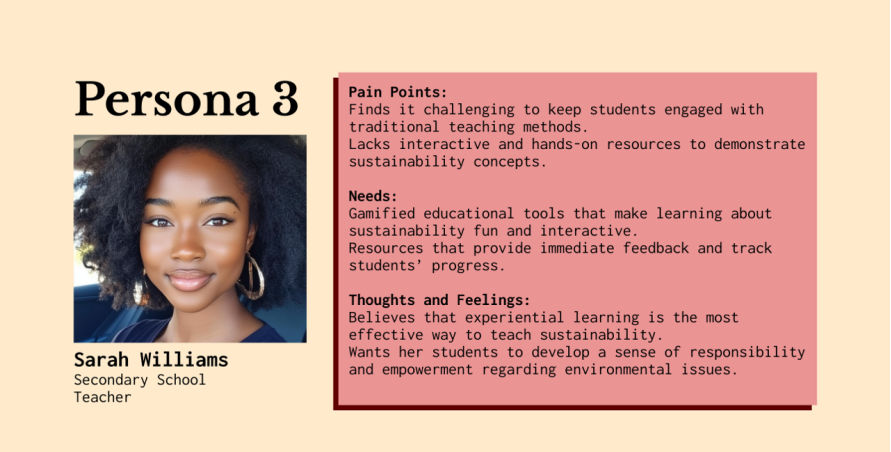

Figure 3: Persona 3 - Sarah Williams, a secondary school teacher

3.2 Problem Statement

A problem statement provides concise description of issues that the project needs to address. It identifies the gap between the current landscape of sustainable development specifically in higher education, with the desired project outcome. By summarising this, the problem statement aims to be a focal reference point when ideating solutions to the insights above.

Students, educators, and university staff are increasingly aware of the importance of sustainability but struggle with engaging, implementing, and maintaining sustainable practices in their daily lives. Traditional methods of sustainability education often fail to inspire and engage, leaving users like Emma feeling overwhelmed and uninspired. Similarly, educators like Sarah Williams seek innovative ways to make sustainability

concepts relevant and exciting for their students, while university facilities managers like Adnan Yilmaz face institutional barriers and resistance when trying to promote green initiatives. Users need interactive, fun, and practical tools that provide clear guidance, achievable goals, and immediate feedback to help them understand and implement sustainable practices effectively.

Despite their commitment to sustainability, these users face significant challenges that hinder their efforts. They need a platform that not only educates but also motivates and empowers them to take actionable steps toward sustainability. Gamification appears to be a promising solution, offering a way to make learning about and practicing sustainability more engaging and effective. By incorporating elements such as social interaction, competition, rewards, and real time feedback, a gamified app can address the diverse needs of students, educators, and university staff, fostering a community of environmentally conscious individuals committed to making a tangible impact. This app must prioritise user centered design, ensuring it is accessible, engaging, and supportive of users' goals to drive meaningful behavior change and overcome the existing barriers to sustainability in higher education.

3.3 Storyboards

Storyboards help translate abstract thoughts into a more concrete format, making it easier to critique and refine ideas. They capture a narrative [26] producing a visual representation of the app's flow. By mapping out the narrative, storyboards make it easier to visualise how our personas will interact with the UI at a high level. This way, we can better insure core user interactions are accounted for and logically sequenced.

Moreover, storyboards serve as a valuable reference point when developing more fine grain designs later on such as use case diagrams and prototypes. Relating storyboards back to the personas enables a deeper understanding of users' thoughts and feelings as they engage with the app. For example, Adnan might be intimidated about where to start after entering the app. Emma might want to collaborate with her friends. Shara might be cornered about the community and culture for her purples.

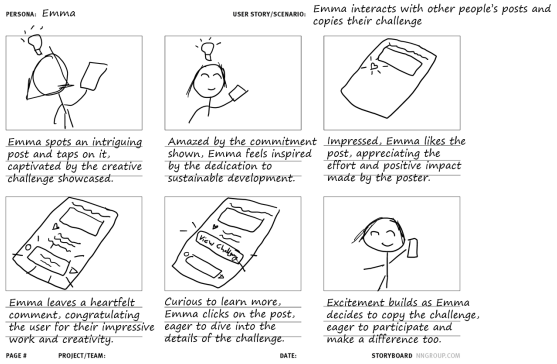

Figure 4: Storyboard 1 - Emma interacts with other people's posts and copies their challenge

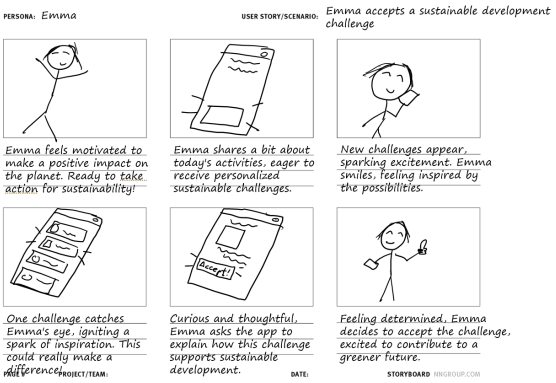

Figure 5: Storyboard 2 - Emma accepts a sustainable development challenge

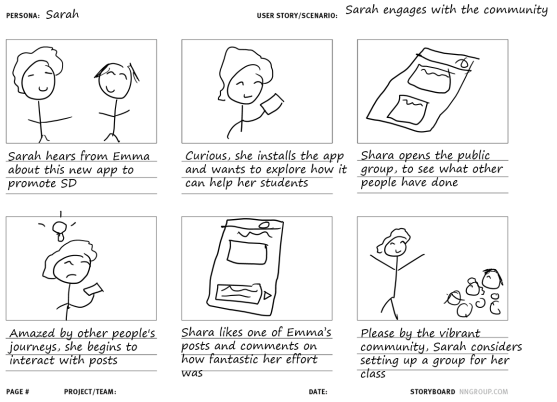

Figure 6: Storyboard 3 - Sarah engages with the community

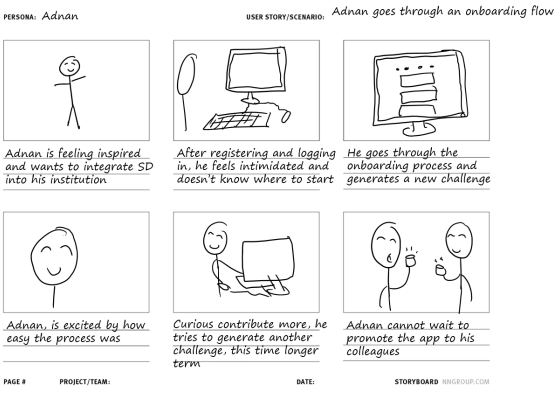

Figure 7: Storyboard 4 - Adnan goes through the onboarding flow

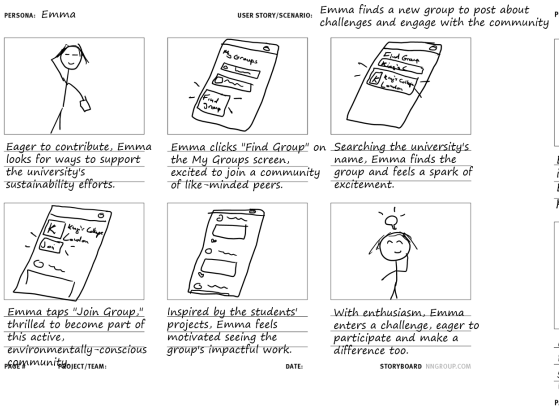

Figure 8: Storyboard 5 - Emma finds a new group to post about challenges and engage with the community

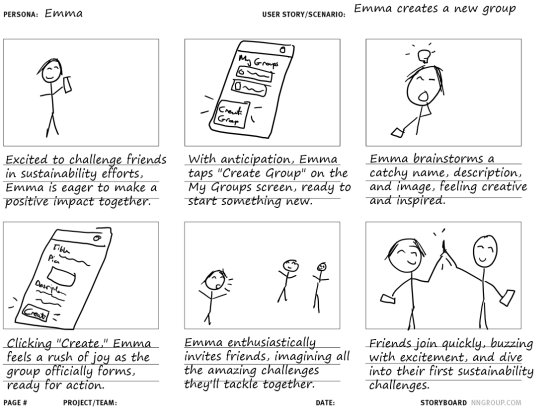

Figure 9: Storyboard 6 - Emma creates a new group

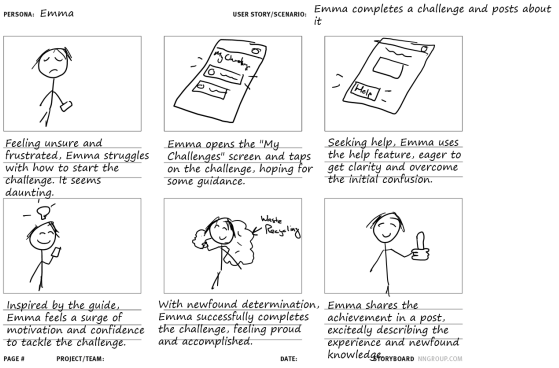

Figure 10: Storyboard 7 - Emma completes a challenge and posts about it

3.4 Ideation and Specification

In the book User Stories Applied, Mike Cohn describes software requirements as a "communication problem" between those who want the software and those who will build it [27]. For an individual project, we can reframe this communication gap by asking ourselves: What are Emma, Adnan, and Sarah trying to tell us? Through personas, problem statements, and storyboards, we have gained insights into who our users are and how we can help them contribute to sustainable development. Now, we can ideate solutions to these problems by creating actionable user stories.

Mike Cohn emphasises the use of user stories to "prevent spreading the decision

making process across the duration of the project" The goal is to gather this information as early and as frequently as possible. To translate our insights into user stories, we employed the "How Might We" (HMW) questions technique, a powerful tool in design thinking and problem solving that reframes challenges into opportunities for creative solutions [28]. This technique starts with a broad challenge and then narrows it down to specific questions, each beginning with "How might we...?" For this project, the HMW technique is particularly useful as it helps identify specific areas where user needs and challenges can be addressed innovatively. The HMW questions draw inspiration from the SHIFT framework, allowing us to explore how sustainable development promotion techniques in the marketing domain can be translated into software engineering.

While it's traditionally recommended to answer HMW questions with unlimited imagination, the project's limited timeframe and resources necessitate a realistic approach. This means the user stories must be specific and centered around the problem statement. To assist with this, I have prioritised the user stories using the MoSCoW technique, categorising them into Must have, Should have, Could have, and Won't have. Compared to other common techniques in software specification prioritisation, MoSCoW among the simplest and quickest to implement [29].

Daryl Kulak and Eamonn Guiney ask, "What do the users see?" [30] Users often view the app as a black box, meaning that from Emma's perspective, they are primarily concerned with "what goes in" and "what comes out." This perspective focuses on the interactions between the user and the app, which, as Kulak and Guiney highlight, is "what really matters in requirements gathering." There are many formats for writing user stories, but the following format in particular is compelling as it captures the interaction and frames the story around the persona: "As a [persona], I [want to], [so that]." This format makes it easier to empathise with Emma, Adnan and Sarah guiding the decision making in a more user centered way, which is important as the user stories will become the reference point as the app's specification.

Finally, for each user story, we have included a set of acceptance criteria (AC). The purpose of the AC is to expand on the abstract user stories and translate them into more system oriented requirements that can be tested. For the ACs, we have again used a user centered format, employing the Given-When-Then technique.

3.4.1 Must Have

These are critical requirements, essential for the MVP

1. **User Story:** As Alex, I want to tell the app about my life, so that I get customised sustainability challenges.
   (a) **AC 1:** GIVEN I want to create a new challenge, WHEN I initiate the process, THEN it should ask if I'd like to provide context about my life.
   (b) **AC 2:** GIVEN I generate a challenge, WHEN the challenge is displayed, THEN it should be relevant to the provided context AND generate an image.

   (c) **AC 3:** GIVEN I select a generated challenge, WHEN I view the details, THEN I should see more information to help me decide.

   - *HMW* provide meaningful challenges based on Alex's daily activities?
   - *HMW* use Alex's daily habits to generate innovative sustainability challenges?

2. **User Story:** As Alex, I want to join a group so I can be inspired by challenges other people have done.

   (a) **AC 1:** GIVEN I am on the groups screen, WHEN I view options, THEN I should have an option to search for new groups.

   (b) **AC 2:** GIVEN I click on a group I haven't joined, WHEN I view group details, THEN I can click to join the group.

   (c) **AC 3:** GIVEN I join a group, WHEN I return to the groups screen, THEN I should see that group listed.

   - *HMW* create a sense of community and inspiration in the groups?
   - *HMW* make group challenges more engaging and inspiring?

3. **User Story:** As Alex, I want to contribute to a group by creating a post so I can share challenges I have done with other people.

   (a) **AC 1:** GIVEN I select an active challenge, WHEN I view the options, THEN I should see an option to complete it.

   (b) **AC 2:** GIVEN I opt to complete a challenge, WHEN I provide a title, image, and description AND post it, THEN a post should be created for all my groups.

   (c) **AC 3:** GIVEN I complete a challenge, WHEN I return to the challenge screen, THEN it should no longer be active AND it should appear in the completed list.

   (d) **AC 4:** GIVEN I view a post I've created, WHEN I select to edit it AND make changes, THEN those changes should be saved.

   - *HMW* make sharing experiences within the group more rewarding for Alex?
   - *HMW* encourage more participation and contributions in groups?

4. **User Story:** As Alex, I want to see a leaderboard in my groups so I can assess how much I can complete with others from my group.

   (a) **AC 1:** GIVEN I enter a group, WHEN I explore options, THEN I should see an option to view the leaderboard.

   (b) **AC 2:** GIVEN the leaderboard loads, WHEN I view it, THEN it should rank users by points earned that week.

- *HMW* create a system that allows users to track and compare their sustainability contributions within a group?
- *HMW* visually represent contributions in an inspiring way?

3.4.2 Should Have

Requirements that can add significant value to the user experience and sustainable development.

1. **User Story:** As Alex, I want to understand how I can accomplish the challenge so that the barrier of inexperience is removed.
   (a) **AC 1:** GIVEN I select an active challenge, WHEN I view it, THEN I should see a help button.
   (b) **AC 2:** GIVEN I click on the help button, THEN I should see advice on how to complete the challenge.

   - *HMW* make learning about challenges an engaging experience?
   - *HMW* reduce the intimidation factor for new users facing challenges?

2. **User Story:** As Alex, I want to understand how the challenge can help with sustainable development so I see how I am contributing.
   (a) **AC 1:** GIVEN I select a suggested challenge, WHEN I view it, THEN there should be a "how does this help" button.
   (b) **AC 2:** GIVEN I click the "how does this help" button, THEN I should see an explanation of how completing the challenge contributes to SD.

   - *HMW* link challenges directly to sustainability goals in a compelling way?
   - *HMW* inspire users by showcasing real world impacts of their actions?

3. **User Story:** As Alex, I want to see new facts about sustainable development so I can broaden my understanding.
   (a) **AC 1:** GIVEN I enter the home screen, WHEN it loads, THEN I should see a section at the top displaying a fact.
   (b) **AC 2:** GIVEN I enter the home screen again, WHEN it loads, THEN I should see a different fact.

   - *HMW* integrate sustainable development education into Alex's daily routine?
   - *HMW* provide interactive and dynamic facts that keep Alex informed?

4. **User Story:** As Alex, I want to learn more about sustainable development so I can broaden my understanding.

   (a) **AC 1:** GIVEN I enter the home screen, WHEN it loads, THEN there should be a "learn more" button navigating to a SD information screen.
   (b) **AC 2:** GIVEN I am on the learn more screen, WHEN I view it, THEN I should see a list of all 17 SD goals with descriptions and images.

   - *HMW* encourage continuous learning about sustainable development?
   - *HMW* make sustainable development knowledge applicable to daily life?

5. **User Story:** As Alex, I want to like other people's posts so that I can give support to others.

   (a) **AC 1:** GIVEN I select a post, WHEN I view it, THEN I can click a like button.
   (b) **AC 2:** GIVEN I liked a post, WHEN I click the like button again, THEN it should unlike the post.
   (c) **AC 3:** GIVEN I like or unlike a post, WHEN the action is completed, THEN I should see a toast message confirming my action.

   - *HMW* encourage more interaction and support within the community?
   - *HMW* show appreciation and encouragement through simple actions like liking?

6. **User Story:** As Alex, I want to comment on people's posts so that I can engage with the community.

   (a) **AC 1:** GIVEN I select a post, WHEN I view it, THEN I can add a comment to that post.
   (b) **AC 2:** GIVEN I view my comment, WHEN I select it, THEN I can delete it.

   - *HMW* make commenting a way to build deeper connections?
   - *HMW* use comments to spark discussions and further engagement?

7. **User Story:** As Alex, I want to receive rewards when I complete milestones so that I can view my achievements.

   (a) **AC 1:** GIVEN I achieve a milestone, WHEN it is recognised, THEN I should receive a toast notification about the reward.
   (b) **AC 2:** GIVEN I enter my profile, WHEN I view it, THEN I can see my reward achievements.
   (c) **AC 3:** GIVEN I select a reward, WHEN I view its details, THEN I can learn more about it.

   - *HMW* make achieving milestones a motivating and rewarding experience?
   - *HMW* create a system that celebrates users' sustainability achievements in a meaningful way?

3.4.3 Could Have

Requirements that will add value to the user experience and sustainable development but can be phased into the app in future iterations.

1. **User Story:** As Alex, I want to receive rewards when I complete milestones so that I can view my achievements.

   (a) **AC 1:** GIVEN I select a post, WHEN I view it, THEN I should see the challenge associated with the post.

   (b) **AC 2:** GIVEN I view an associated challenge, WHEN I select copy, THEN it should become an active challenge for me.

   (c) **AC 3:** GIVEN I enter the my challenges screen, WHEN I view it, THEN I should see that challenge listed as active.

   - *HMW* encourage the sharing and replication of successful challenges?
   - *HMW* use popular challenges to inspire more participation?

2. **User Story:** As Alex, I want to be notified when others engage with my posts, so that I'm up to date.

   (a) **AC 1:** GIVEN someone likes or comments on my post, WHEN it happens, THEN I should receive a notification indicator on my profile image in the header.

   (b) **AC 2:** GIVEN I click on my profile image, WHEN I do so, THEN I should see an overlay with all my notifications.

   (c) **AC 3:** GIVEN I view the notification, WHEN I do so, THEN the indicator should be cleared.

   - *HMW* keep users informed about interactions with their posts?
   - *HMW* keep users informed and engaged without overwhelming them?
   - *HMW* use notifications to build a sense of real time community interaction?

3. **User Story:** As Alex, I want to create a group so I can collaborate with my peers.

   (a) **AC 1:** GIVEN I enter the my groups screen, WHEN I view it, THEN I should see an option to create a group.

   (b) **AC 2:** GIVEN I select to create a group, WHEN I fill in the form and submit, THEN the group should be created.

   (c) **AC 3:** GIVEN I return to the my groups screen, WHEN I view it, THEN I should see the new group listed.

   - *HMW* facilitate private collaborations among users?

- *HMW* create a private and collaborative space for focused group activities?
- *HMW* make private boards a tool for deeper collaboration and innovation?
- *HMW* balance privacy with community engagement?

4. **User Story:** As Alex, I want to view a tutorial so I can get up to speed with how to use the app.
   (a) **AC 1:** GIVEN I access the home screen for the first time, WHEN it loads, THEN I should see a tutorial indicator.
   (b) **AC 2:** GIVEN I select the tutorial indicator, WHEN I view it, THEN I should receive information about different features of the app.
   (c) **AC 3:** GIVEN I complete the tutorial, WHEN I return to the home screen, THEN I should no longer see the indicator.

   - *HMW* make onboarding an intuitive and engaging experience?
   - *HMW* ensure that users quickly understand the app's features without feeling overwhelmed?

5. **User Story:** As Alex, I want to see an onboarding screen so that I am set up with a challenge right away.
   (a) **AC 1:** GIVEN I log in for the first time, WHEN I access the app, THEN I should see an onboarding screen that allows me to view groups AND accept a challenge.
   (b) **AC 2:** GIVEN I log in again after completing onboarding, WHEN I access the app, THEN I should be taken to the home screen instead of the onboarding screen.

   - *HMW* streamline the onboarding process to quickly engage new users?
   - *HMW* provide a seamless transition from onboarding to active participation?

6. **User Story:** As Alex, I would like to speak with the app so I can exchange information.
   (a) **AC 1:** GIVEN I choose to generate a new challenge, WHEN I view the options, THEN I should see a microphone button.
   (b) **AC 2:** GIVEN I select the microphone button, WHEN I speak, THEN I can use voice input instead of typing to give context to the challenges.

   - *HMW* make speaking with the app a natural part of the user journey?
   - *HMW* leverage voice interaction to enhance user engagement?

3.4.4 Won't Have

Requirements, while adding value, are out of project scope. This can be due to time feasibility or not being relevant enough for the project.

1. **User Story:** As Alex, I want to post video content to the board so others can see my SD journey.
    - *HMW* use video content to inspire and educate the community?
    - *HMW* make it easy to create and share impactful video stories?

3.5 Use cases

Earlier, from the storyboard, we gained better clarity of the user journey as Emma, Adnan and Shara complete various user stories. By mapping these interactions from a visual user perspective to a broader systems view, considering both system actions and actors, we can highlight additional functional requirements beyond the just user stories and acceptance criteria. This is where use cases come in.

Daryl Kulak and Eamonn Guiney [30] recommends focusing use cases on business interactions as part of their guiding principles. In this implementation, the diagrams encapsulate the user stories, providing a structured way to visualise and connect different user actions and experiences. This approach means that the use case diagrams naturally extend from the specification. We gain a holistic view of the app ensuring different aspects of the user experience are considered. This is especially useful when considering an appropriate software architecture later on.

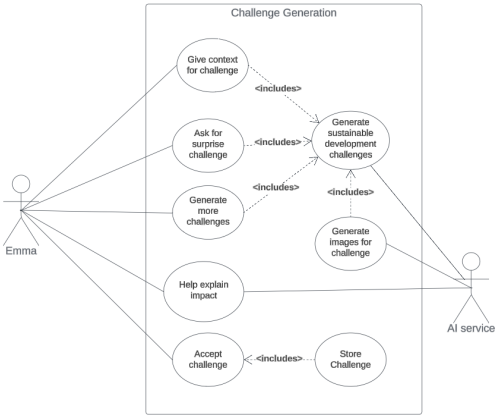

Figure 11: Use case diagram: Generating a challenge

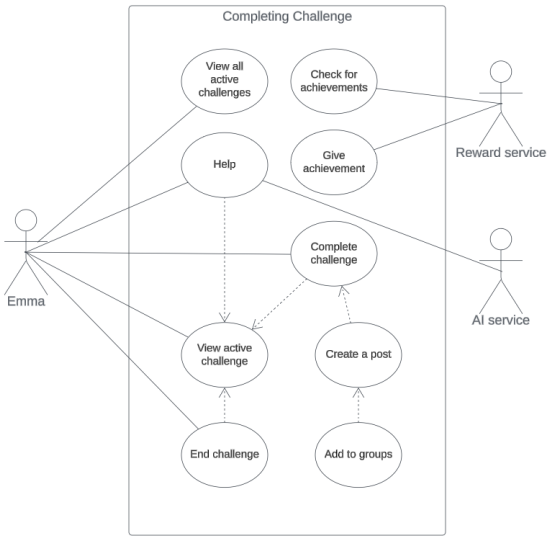

Figure 12: Use case diagram: Completing a challenge

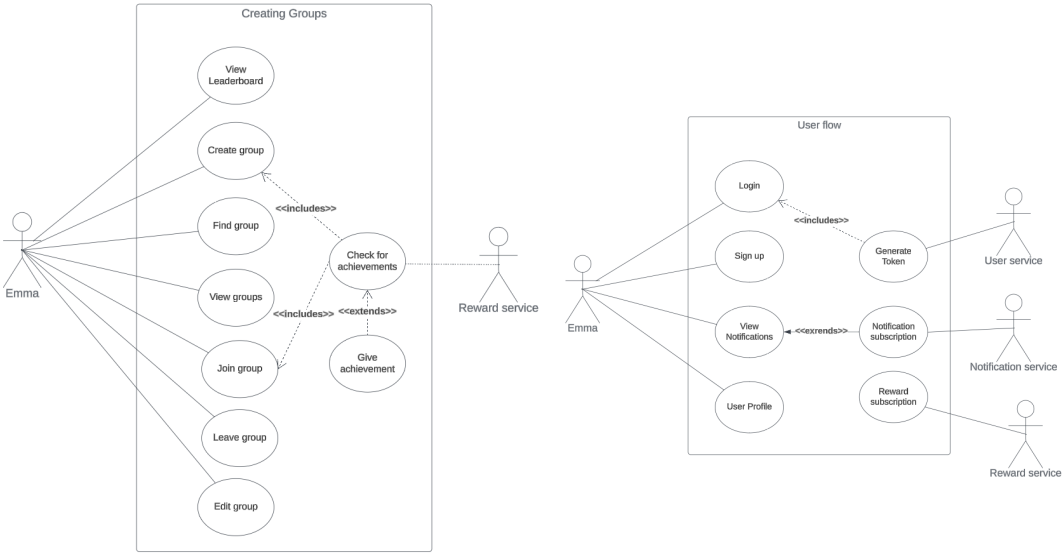

Figure 14: Use case diagram: Completing a challenge

Figure 13: Use case diagram: Generating a challenge

Figure 15: Use case diagram: Generating a challenge

3.6 Low fidelity prototypes

Low fedility prototypes are vague in nature and minimally formed in attributes [26]. They provide a cost effective environment for experimentation, allowing quick iteration of ideas without the pressure of investing significant time or resources. This flexibility is crucial for exploring different design solutions to solve the complex challenges presented by the user stories. For example, how can we create a UI which allows Emma engage with other like minded sustainable development advocates? The storyboards and use case diagrams created earlier add significant value to the low fidelity prototypes. By having already established the flow, we can concentrate on the finer interactions, speeding up the prototyping process.

The prototyping tool used is Figma. One key benefit of this tool is the ability to create components, which allows for reusable elements. This is particularly useful when transitioning to Frontend development, as it aligns with component based architecture. In UX design this is commonly referred to as the "Design System". Often, low fidelity prototypes are mocked up through sketches, our implementation can be seen as closer to mid fidelity. However, purpose that is, fast exploration and experimentation, still holds up due to my experience with using Figma. Otherwise, pen and paper might be more suitable. Figma's powerful features for mocking interactions enable us to simulate user flows and interface behavior, providing a clearer vision of the user experience.

Figure 16: Low fidelity prototype

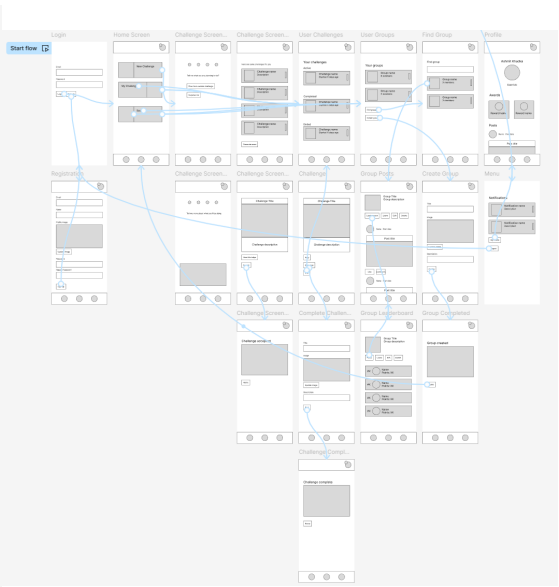

Figure 17: Low fidelity prototype, interactions

3.7 High fidelity prototypes

After experimentation and exploration, we can proceed to replicate the final user interface using high fidelity prototypes. This phase involves planning detailed design decisions, such as color schemes, typography, and responsive components . Typically, we see this technique employed as a communication tool by designers, developers and stakeholder for agreeing on the final user experience. [26]. For this individual project, creating a detailed and accurate representation of the final product helps eliminate ambiguities, making it easier to translate designs into code . Moreover, It allows for meaningful feedback users through usability testing without the expenditure of coding. Figma's sharing tool allows us connect with people such as Emma, Adnan and Sarah so we can fix faults in the UX at more flexible stage.

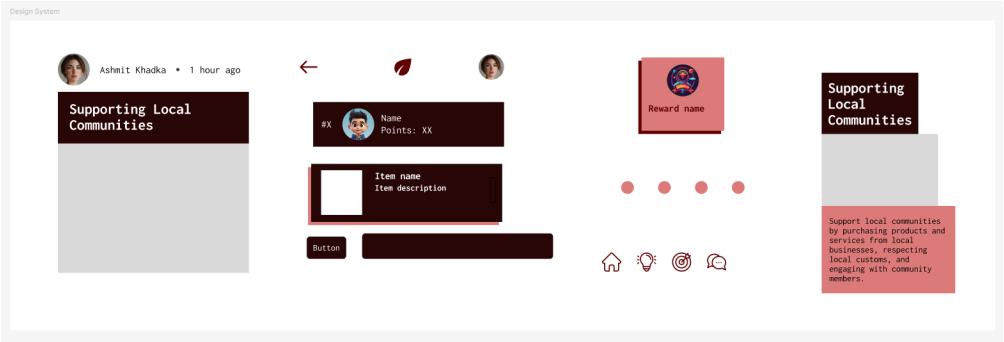

Figure 18: High fidelity prototype - design system

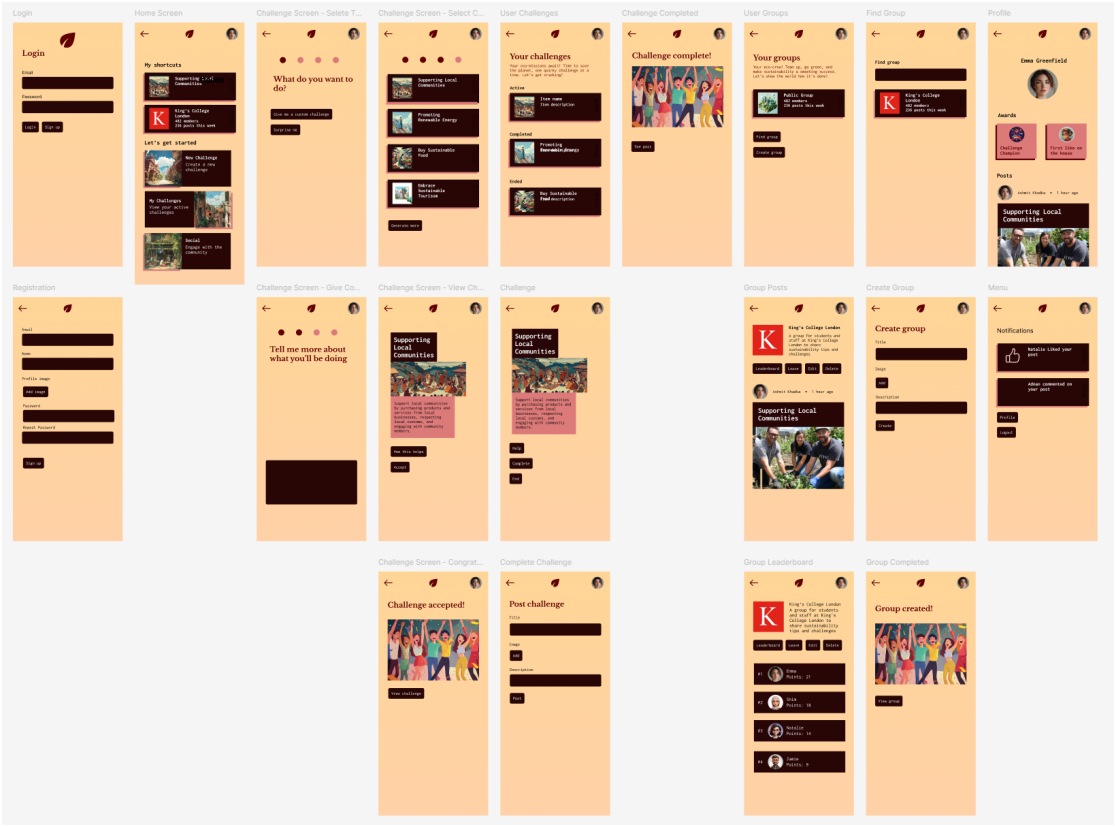

Figure 19: High fidelity prototype

A mobile first design principle is employed for the prototypes as this is the target user platform where we anticipate the majority of the user base to be. When transitioning to a more larger view port such as a desktop or table, we aim to utilise more of the horizontal screen real estate by moving the navigation to the left and breaking down the screen content into multiple columns. This means the user is able to see more of the content at once.

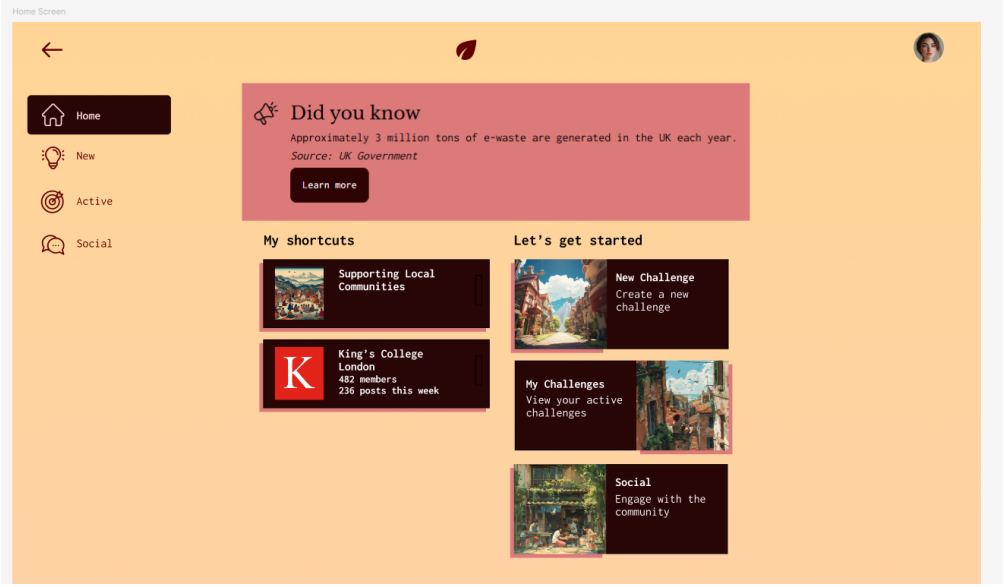

Figure 20: High fidelity prototype - desktop view

3.8 Architecture

If we want to make an impact, we need as many people as possible to use the app; therefore, it is important to consider performance and scalability as intrinsic aspects of the this softwar engineering problem. In this project, we will be using the VIPER architecture, a clean architecture pattern designed to improve the modality and maintainability of the app. VIPER is an acronym for View, Interactor, Presenter, Entity, and Router where each component has a distinct role in managing the app's functionalities. This architecture is well suited for handling complex interaction requirements, ideal for feature set that contains items such as AI integration, social media features and other interactive modules. By separating concerns into distinct layers, VIPER facilitates independent development, testing and maintenance of these logic areas.

1. **View:** Responsible for displaying data to the user and capturing user interactions.
2. **Interactor:** Containers the business logic, making it central to handling tasks like challenge generation and user reward distribution.
3. **Presenter:** Fetches data from the Interactor and prepares it for the View.
4. **Entity:** Entities are simple data structures used to transfer data within the app. They help in managing user data. such as user posts and group information.
5. **Router:** The Router manages the app's navigation flow.

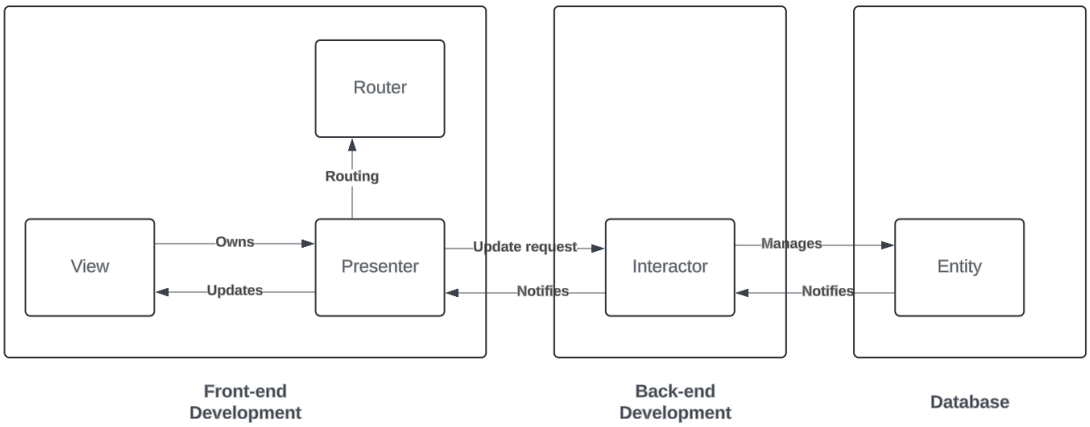

Figure 21: VIPER architecture overview

Compared to other architectures like MVC (Model-View-Controller) or MVVM (Model-View-ViewModel), VIPER offers a more granular separation of concerns. MVC can sometimes blur the lines between business logic and UI logic, leading to complex and tightly coupled code. MVVM, while more modular than MVC, still mixes some business logic with presentation logic in the ViewModel. VIPER's strict division helps in keeping the codebase clean, making it easier to manage and scale. The down side is we get a more complex architecture which can introduce communication overhead between the layers.

3.9 Testing

To create testable software, we must plan for visibility and control of components at the design level [31]. Principles followed in this project when designing software components include tagging elements with test IDs, decoupling dependencies using interfaces and dependency injection, using the observer/blackboard pattern and as mentioned with VIPER, ensuring separation of concerns among modules. Test coverage for individual components and their granular requirement can be handed by unit tests. While, more holistic user story testing can be performed using integration tests. We will talk more about testing in the implementation chapter.

4 Methodology and Implementation

Implementation has been split into three sections: Frontend, Backend, and Database, as these are the main logical areas of separation. Before writing a single line of code, we have to plan the technology foundations to build a reliable, scalable, and high performance application

4.1 Technology Stack

This app will be based on the MERN (MongoDB, Express, React, Node) stack.

- **MongoDB:** A NoSQL database used to store and manage data in a flexible, JSON-like format.
- **Express.js:** A backend web application framework for Node.js that simplifies building APIs and web servers.
- **React.js:** A frontend JavaScript library for building user interfaces, particularly single page applications, using a component-based architecture.
- **Node.js:** A JavaScript runtime that allows you to run JavaScript on the server side, enabling backend development with JavaScript.
- **Router:** The Router manages the app's navigation flow.

The MERN stack uses JavaScript across the entire application, from the database (MongoDB) to the server (Node.js and Express.js) and the frontend (React). This unification simplifies the development process.

React and Node.js excel at building highly dynamic, single page applications (SPAs), which are crucial for the app's interactive and gamified elements. The stack supports real-time data flow (individual UI updates) and user interaction, aligning with needs such as social engagement and challenge generation.

MongoDB's schema less nature allows for flexible data models, which is ideal for scaling the app as it evolves. The scalability of the MERN stack ensures that the app can handle increasing loads and complex features without significant architectural changes.

Extending from the architectural design, there are the same three layers as show in the architectural diagram, this time from a technology stack perspective.

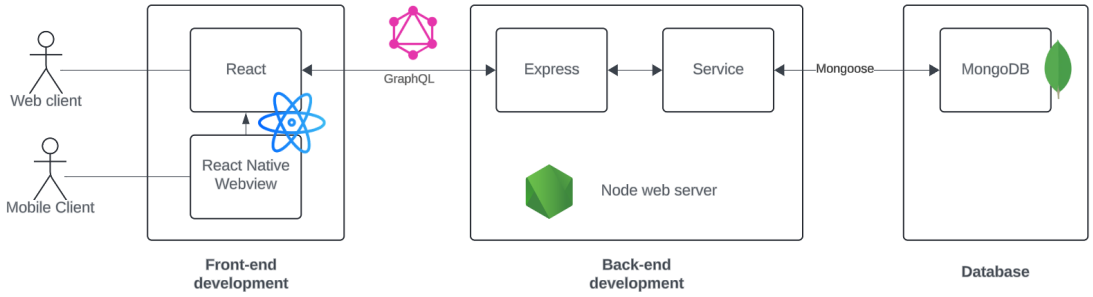

Figure 22: Tecnnology stack overview

Compared to alternatives like Flutter with Dart and .NET with MAUI, the MERN stack offers distinct advantages:

- **Flutter** is great for cross platform mobile development but falls short in web application support and requires knowledge of Dart, a less popular language than JavaScript. For a project centered around complex web interactions, MERN provides a more suitable environment.
- **.NET with MAUI** offers powerful cross platform capabilities for developers familiar with C#, but it can be resource intensive and less suited for building highly dynamic SPAs. The flexibility and performance of React in handling SPAs make the MERN stack a better choice for our needs.

A shortcoming of the MERN stack for this project is its performance on mobile native features, where Flutter might excel. To overcome this, we can use web views to integrate mobile specific features while maintaining a consistent user experience across platforms. This approach is somewhat of a compromise in performance; however, by abstracting backend tasks away from user devices to backend servers, it minimises the impact of using the MERN stack for the app. From the low fidelity prototypes and user stories, there is little benefit to using the native features of Flutter.

The MERN stack complements the VIPER architecture by enabling a clear separation of concerns. Each component of VIPER—View, Interactor, Presenter, Entity, and Router—can be implemented as distinct modules within the MERN framework. For instance, React components align with the View and Presenter layers, while Express.js and Node.js can handle the Interactor's business logic. This modularity supports maintainable and scalable code.

4.2 Frontend

The frontend is an essential element to get right for this project. To tanslate the complex elements envisioned in my prototypes, I've had to use several techniques to insure I create dry code. In this secrion, I'll split the frontend into 3 sections, View, Presenter and Routers based on the elements of the VIPER architecture. But first, Ill motivate the use of React.

4.2.1 React

React is a robust frontend framework developed by facebook and used in production today by some of the world's largest organsiations such as Meta, to build bring to life some of the most sophisticated UX such as Instagram, Netflix, ArBnB. It's reliability and scalability is proven at the highest level and used by billions of people daily.

React offers best in class community support. As highlighted before, JavaScript brings together the community like no other language which millions of libraries and frameworks. Of all the frameworks, react has the most extensive set of external libraries [32].

Performance is a key consideration, the use of React's viertual DOM improves application with firequiet UI updates. This is key to delivering an engaging experience for features such as live notifications, liking post and sharing comments with people's

challenges. With regards to render times, Vue comes out on top but React is not far behind [32].

While React's Simplicity and flexibility allows for faster development, a common pitfall is writing code which head towards poor design patterns. This is where frameworks such as Angluar excels at by providing defined and reliable was solving challenges. To overcome this, I've referred to the SOLID principles to write maintainable code which Ill elaborate more later. Another major concern of using React was it's third party librarties. These can introduce security risks and dependnencies on code when might no longer be maintained long term causing scalability issues later on. For this I've made use of services such as Snyk to review vulnerabilities of packages and vet out high risk third party libraries. Using the the NMP web site, I can also review packages activity history to insure that the libararies Im using is well maintained. Overall, dusp

4.2.2 VIPER - View

The view utilises a componentisation pattern, a natural extension from the prototypes created using Figma. By breaking down the screen into components, areas of concern are separated, resulting in more maintainable code that allows for code reuse. For example, the challenge view component from the "My Challenge" screen can be reused in the homescreen to provide a shortcut to the user's most recent accepted and active challenge.

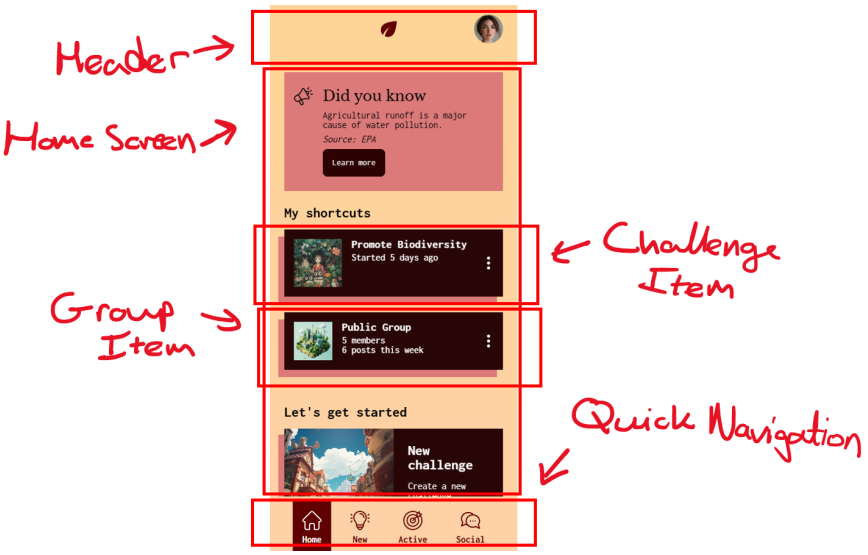

Figure 23: Componentisation of UI elements in the Home screen

Here is an example of how componentisation is used in the homescreen. Different elements such as the app header, app footer, challenge item, group item, and activity tiles are all combined and reused to provide a unique set of information and actions that meet the requirements for the home screen.

A limitation of using React with web view is we no longer have access to native mobile UI elements such as drawers and tabs. To overcome this we can intergrate custom libraries in the view resolve complex UI problems by reusing reliable and well documented code. For example, f4rz1n's excellent react-modern-drawer library allows a drawer overlay to appear when triggered through user interaction. It means Emma can view actions for her active challenge from the home screen shortcuts without navigating to the challenge screen. Integrating custom libraries into the app requires thoughtful planning and understanding of the library. For example, in the drawer, we must plan trigger points for the overlay, custom styles and actions to intergrate semlessly with the app.

Similarly, sniphpet's popular react-toastify library allows toast overlays to appear throughout the app. This an example of how custom libraries allows the UI to go beyond native mobile functionality as this pattern is not included with iOS and Androids. The toast pattern enhances UI feedback of actions so for example, Emma spots that she had liked a post. It can also facilitate error prevention, for example if Shara accidentally deletes a post in a group she's created for her class, she can restore her pupil's post.

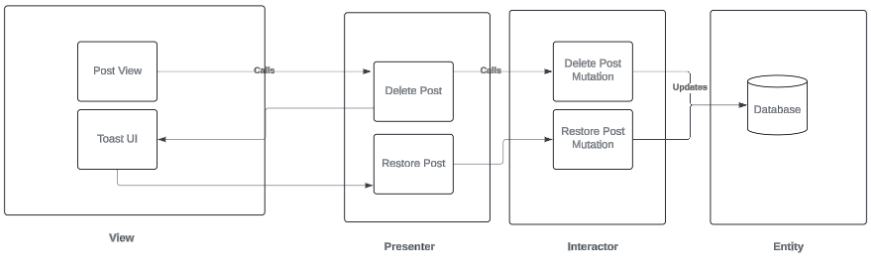

Figure 24: Overview of undo action through toasts

Another example is bluebill1049's react-hook-form. This enables dynamic form validation for input fields consumed by screens such as user registration, challenge completion and group creation screens. Doing so, we can use constraints to guide Sarah towards the happy path and prevent errors. Impementation involves intergrating the library logic into the control components. For the textbox, we show errors if there are validation errors and disable the button to prevent incomplete data being sent. Again, this is another example of how using React libraries enables the project to go beyond native UI capabilities of IOS and Android compared unlike MAUI or simmilar technology stacks.

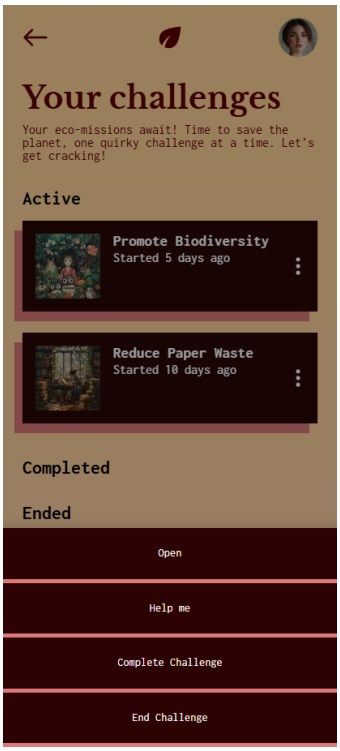

Figure 25: Drawer pattern

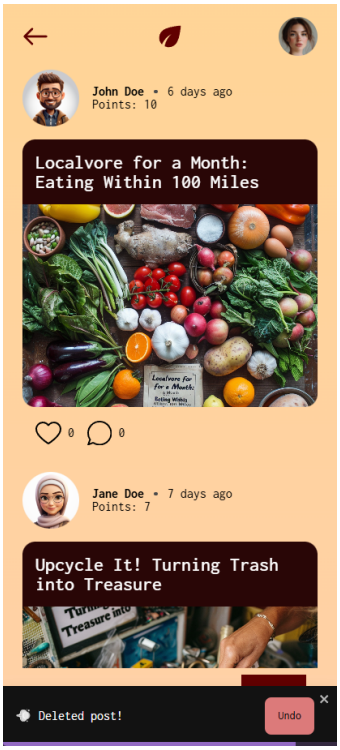

Figure 26: Toast message pattern with undo button

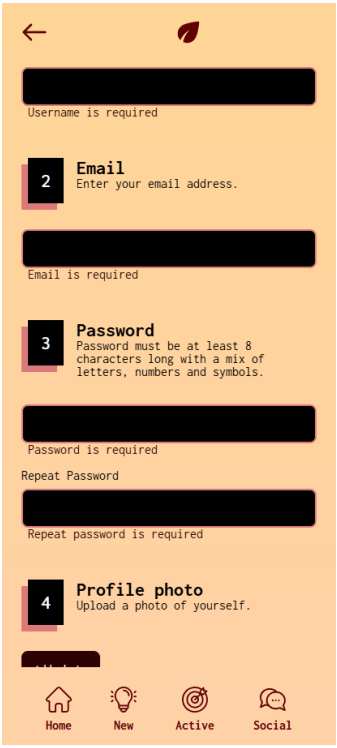

Figure 27: Form validation

4.2.3 VIPER - Presenter

The job of the presenter layer is to maintain the dependency inversion principle between the view and the interactor. In React terms, the custom hook pattern are utilised to solve this challenge. Custom hooks are essentially serivce files for views, encapsulating logic that can be reused across multiple components. For example, a custom hook called useGroupActions can decouple group logic from the views. Such logic includes loading group data, joining groups, creating groups, etc. By implementing the custom hook as pure functions, we insure consistent output over different use cases. This keeps the view simple, easier to maintain and more testable. Now a single join group function can be referenced by multiple components, such as the group item drawer and the group screen.

The blackboard architecture in React, using useContext, enhances the user experience by centralizing and simplifying state management. For example, user data can be retrieved once after login and stored in a “cache” for reference later. This setup ensures consistent and synchronised data across the application, reducing discrepancies and decoupling logic as apposed to using other techniques such as prop drilling. We can set the data one and have multiple components listen for updates. Storing data this way also significantly reduces API calls, which improves app performance and server side resource

utilisation. Other areas where the blackboard pattern is used includes setting buttons to activate the drawer, setting text for toast messages and passing prompts to the guide screen for AI assistance.

Users benefit from seamless interactions, as data updates are efficiently shared across components. For example, by setting a drawer state with button information, a drawer can be rendered. This approach also accelerates development and maintenance. By providing a single source of truth, it enhances the overall performance and predictability of the app, ensuring a smoother, more intuitive experience for users. of the app, ensuring a smoother, more intuitive experience for users.

4.2.4 VIPER - Router

Given the user stories of the app, managing routes can become a complex challenge. Fortunately, many libraries offer excellent support for handling routing in a scalable way with React. For this project, I've used the popular react-router library, which provides numerous useful features. These include dynamic route matching, nested routing, and route transitions. The state feature, in particular, is used throughout the app to provide context to the next screen. For example, when Emma selects an active challenge from the "My Challenges" screen, we can pass in that challenge ID to the challenge screen so the correct data is retrieved from the server side. One notable feature that is absent from the standard react-router package is protected routes. To address this, I've engineered a custom solution to create private routes by developing a wrapper component. We need to ensure that users are authenticated before accessing sensitive screens such as "My Groups" and "My Profile." The wrapper component, called ProtectedRoute, handles this by checking for a JSON Web Token (JWT) in the local storage and decrypting it to obtain the user data. This data is then passed through a context provider, allowing Adnan to keep his session state across different components and routes.

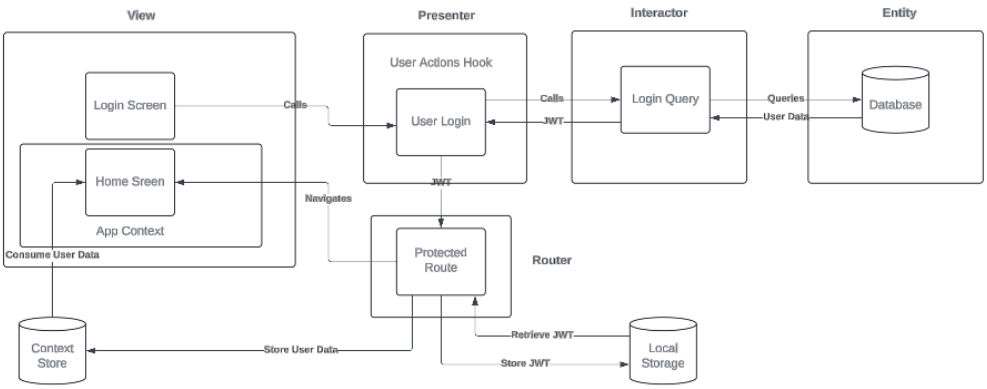

Figure 28: Overview of protected routes

Using JWTs in this manner keeps the app secure by ensuring that only authenticated

users can access certain routes. JWTs are a secure way to transmit information between parties because they can be digitally signed. By verifying the token, the app can confirm the identity of the user, protecting sensitive data.

4.3 Backend

backend architecture comprises three core technologies: Express, Node.js, and GraphQL.

- Express: This is a lightweight web framework for Node.js that facilitates handling HTTP requests. It sets up the web server and communicates with the GraphQL models and services. Additionally, Express handles middleware, such as CORS (Cross Origin Resource Sharing), to protect against security issues like those listed in the OWASP Top 10 vulnerabilities.
- Node.js: A JavaScript runtime built on Chrome's V8 engine, Node.js enables server side scripting and runs the backend logic for web applications.
- GraphQL: An open source query language for APIs that enables clients to request specific data. GraphQL has three core features: queries, mutations, and subscriptions.
  - Queries: Similar to GET requests in REST APIs, they retrieve data from the database. For example, a query like postsByUser fetches all posts for a specific user ID.
  - Mutations: Comparable to POST, PUT, and DELETE requests in REST, mutations modify data. For instance, the createChallenge mutation takes challenge data as parameters and creates a challenge object in the database.
  - Subscriptions: These listen for data changes and notify the frontend in real time, such as alerting users to new notifications. Use of GraphQL and comparison with RESTful APIs

GraphQL offers several advantages over traditional RESTful APIs for managing requests from the frontend. For this project, the most significant benefit is GraphQL's structured nature, which provides a well proven and robust way of handling requests. RESTful APIs rely on principles rather than strict structure, making it easy to lose control and prone to poor code practices. For example, multiple teams writing RESTful APIs for their microservices often use unique formats, leading to inconsistent and difficult to read code in the frontend layers. This results in a messy and inconsistent codebase, making it challenging to program when updating dependencies from other teams' applications. GraphQL addresses this issue by providing a defined schema for data and endpoints. All endpoints are handled in a predictable and defined manner, leaving little room for interpretation. This consistency simplifies the integration process and improves the maintainability of the codebase. This is an important consideration in order to build using scalable technology. A controlled experiment into development

time reveals that developers can complete the implementation of identical tasks more quickly using GraphQL. [33].

Another key benefit for this project is that GraphQL allows more control over the data requested in the Frontend. For instance, we can set up a user model containing all user information, such as password data, and retrieve only the necessary non sensitive information. This approach eliminates the need to store sensitive data in separate tables, avoiding a scattered database structure. For Shara, she can be assured that her sensitive personal information won't be sent over the school network when accessing tivial user data. This improves security and again reduces the development effort needed to create more granular APIs as is the issue with REST APIs.

The subscription feature is another significant motivator for using GraphQL in this project, as it extends beyond the capabilities of traditional REST APIs by enabling the implementation of an observer pattern. For example, when someone likes Emma's post, we can push a real time notification to her active session, ensuring she remains informed about her engagement with the sustainable development community. Another practical application is in real time rewards. When Adnan completes an activity, such as finishing a challenge for the first time, we can validate his achievement on the backend and, if a reward is earned, instantly push a toast message to alert him.

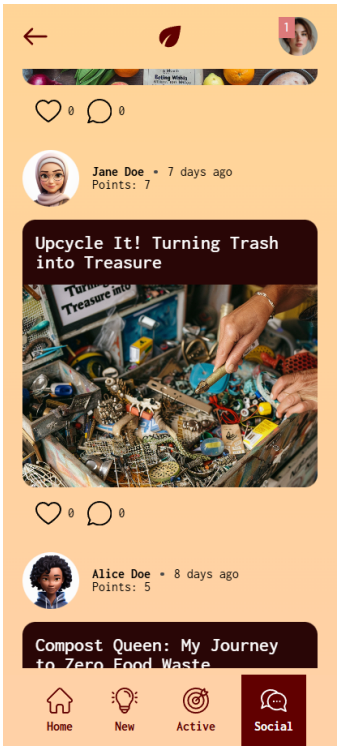

Figure 29: Real time notification indicator in header

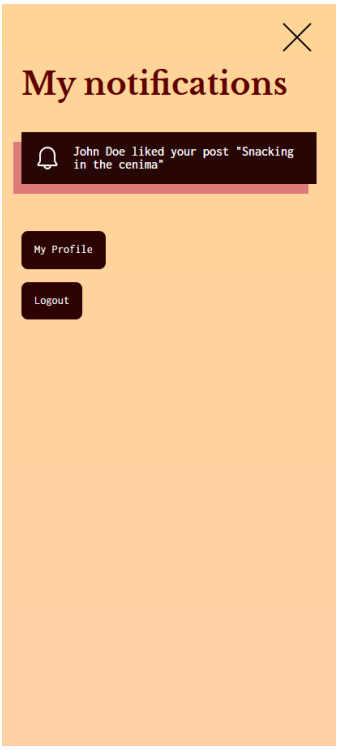

Figure 30: Real time notification push in overlay menu

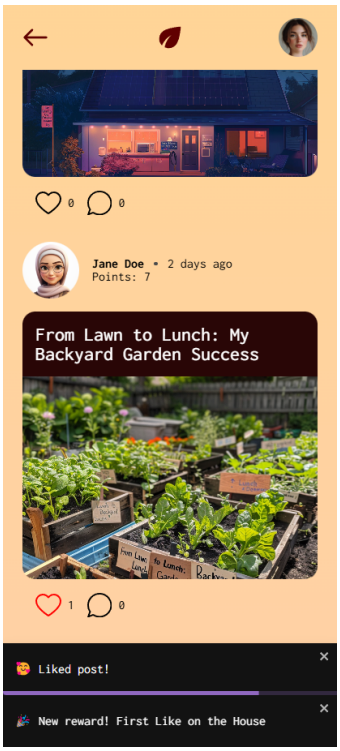

Figure 31: Toast message when a reward is unlocked

4.3.1 VIPER - Interactor

In the VIPER architecture, the Interact handles the business logic. For this, I have created separate service files, decoupling GraphQL logic from MongoDB database logic, thereby following the DAO (Data Access Object) architecture. This separation maintains the dependency inversion principle, allowing for more modulator architecture. For example when migrating database technology (e.g., from MongoDB to SQL), Emma will have minimal down time in the platform as the substitution won't affect upstream code. Furthermore, I've organised the flow for each model at all levels—service, query, mutation, and subscription layers—enabling a microservices architecture. This approach benefits scalability; for instance, user posts, which have more activity, can be hosted on a more capable server to optimise performance and cost. This architecture also allows for dedicated teams to handle specific functionalities, similar to how my current organisation operates, where a separate team manages employee management for payroll software. Such considerations are important design decisions to make to insure that the performance and maintenance both scales with user base and complexity of additional user stories.

Allowing Emma to add images to her profile, groups, and to generate imagery for challenges requires a storage solution. Amazon Web Services (AWS) offers S3 buckets highly regarded for their excellent documentation, robust APIs, and proven resilience, trusted by some of the largest organisations. The cost structure of S3 storage allows for scalability as the user base grows.

4.3.2 AI

Using generative AI breaks many of the barriers stated in the problem statement. Through recent breakthroughs in generative AI, a personalised experience that tailors to the needs of the user can be created. Doing so creates more engaging experiences, enabling Emma, a busy univercity student, to better integrate sustainable practices into her daily life. For this project, the AI is based on OpenAI's ChatGPT LLM and DALL-E image generation platforms, with Reggie, the King's College London mascot, as the AI persona. OpenAI was chosen over others due to its comprehensive feature set that includes both LLM and image generation while also boasting excellent developer documentation.

ChatGPT facilitates the generation of customised user challenges as the user provides context about their day. Several techniques are employed to ensure responses of sufficient quality. Firstly, to set the tone and language, the base prompts pattern starts with: "You are a sustainable development goal expert. People come to you for advice on how to help contribute towards the sustainable development goals. These are mostly students." This explains what kind of help is required by Emma. Next, we specify the format we need in the response. Finally, a concrete example of the response format is provided to ensure the response is in JSON. These steps ensure relevant challenges with the correct language and information are received, which can be parsed and understood by the app. Part of the response generates a prompt that DALL-E can consume for creating an image for the challenge. This results in a powerful self contained multi-agent system.

Similarly, Reggie can also offer a chatbot like experience where different trigger points automatically send automated prompts to help the user. For example, when Adnan selects the "How does this help?" button on the challenge screen, Reggie explains to Adnan the impact of the challenge towards sustainable development. Another example is the "Help me" button on the user challenge screen, which guides the Adnan with suggestions he can use to complete his challenge. Again, this breaks many of the barriers stated in the problem statement and tackles feelings of being overwhelmed and uninspired when trying to contribute to sustainable practices. Agbedahin [2] highlights how education around sustainable development is central to addressing the engagement gap. Through AI, we can equip Adnan with knowledge and skills required to act sustainably.

AI is also used for profanity checking before saving comments on posts. The backend accepts the comment as a prompt to determine if the language used is defamed safe. Using AI this way helps Sarah build a friendly community of sustainability conscious students for her classroom group. Together this demonstrates the versatility of LLM being used for generating content, as an education tool, and for moderating content.

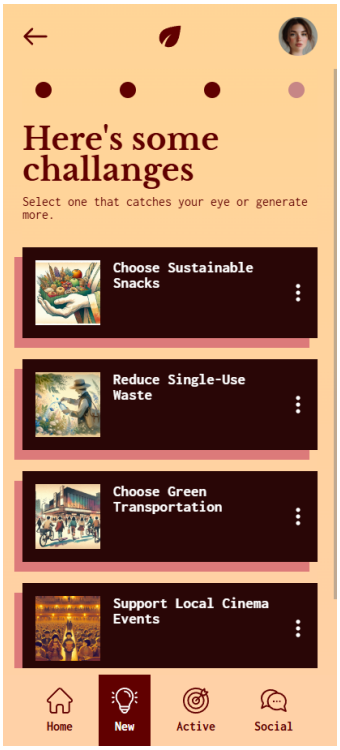

Figure 32: AI generated sustainable development challenges

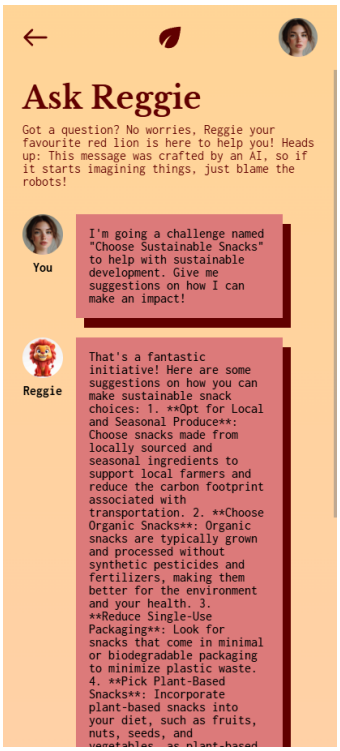

Figure 33: AI chatbot for help

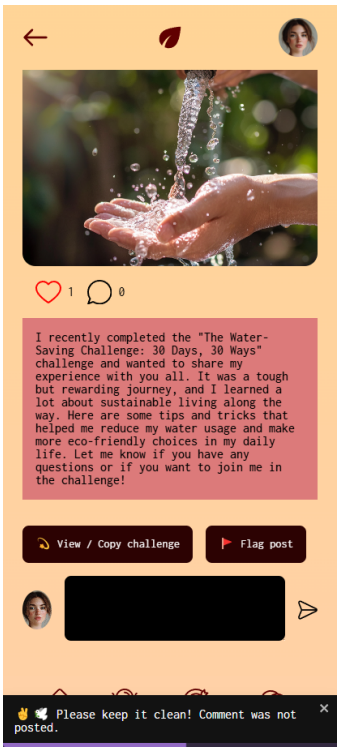

Figure 34: AI moderation for comments

Despite basing the AI on the OpenAI platform, through the DAO (Data Access Object) architecture, the platform implementation can be substituted without any upstream side effects to the frontend or database. This abstraction keeps the app modular, protecting it from volatility in API price and performance.

The four main challenges of generative AI in this project are getting quality responses, staying relevant to the topic, managing API cost and hallucinations. For the first challenge, the base prompt pattern is used to get quality, predictable results. For topic relevance, user input is minimised to specific prompts triggered by the actions mentioned above. This also helps with managing API cost while still adding value to the experience. Hallucinations are responses that are inaccurate, misleading, or entirely fabricated. This is an inherit problem with using LLM, therefor, we must warn the user by adding a message above Reggie's response 33.

4.3.3 Database

For this project, a NoSQL database, MongoDB, is used. Typically, in most scenarios, SQL databases tend to outperform MongoDB. A study comparing MongoDB with Post-

greSQL, both hosted in the same spec VPC, found the average response time to be almost half of PostgreSQL when compared to a MongoDB cluster in a set of spatio-temporal queries [34]. Despite this, NoSQL databases provide a set of compelling benefits that specifically help this project. As the article elaborates, with performance optimisations using indexing, response time differences can be drastically reduced, creating an even playing field.

MongoDB's schema less design allows for flexible data structures, making it easier to adapt to evolving requirements without the need for complex schema migrations. This flexibility is crucial for a project that aims to integrate diverse features such as AI driven recommendations, social media interactions, and gamification elements. Using the low fidelity prototypes to derive the initial schemas proved to be an effective strategy, but more data was required, which could only be revealed at later stages in engineering. For example, adding a status property to challenges to signify when a challenge has been completed. The schema less architecture allowed for faster changes and development. This will also be relevant in future iterations of the project if users want to add more features.

MongoDB can store and manage various data types, including structured, semi structured, and unstructured data. This capability is essential for handling the diverse data generated by the app. For example, user and post IDs can be stored in an array as properties for groups. This removes the need to create composite tables as with SQL, which declutters and simplifies the database.

MongoDB supports horizontal scaling, meaning it can distribute data across multiple servers, enhancing the application's ability to handle increased loads and ensuring high availability. With many users, each producing multiple artifacts such as posts and challenges, the potential for the database object count to expand exponentially is high. The ease of scaling makes it an ideal future thinking technology compared to SQL databases. In fact, leveraging MongoDB's cloud services, Atlas, scaling will be managed by the infrastructure. For this project, the Replica set feature is used to create a duplicate secondary server instance of the database. Doing so provides better redundancy against network partitions and system failures which is important as in production, the database will be hosted in a VPC.

Figure 35: MongoDB Replica Set Configuration

MongoDB's rich query language and powerful aggregation framework allow for complex queries and data processing, making it easier to generate insights and reports. This is one of NoSQL's main shortcomings when compared to SQL databases, as it cannot

match the query features of SQL. MongoDB, however, mitigates this to a large extent using indexing. For the project, mongoose, a Node library, is used to interact with the database and run queries.

Despite this, MongoDB databases inherently do not support transactions and joins. This is a disadvantage as the app requires complex queries across multiple collections. However, with the group example, MongoDB's structured data capabilities are utilised to store related data such as user IDs for a group object as an array. Now we are able to speed up query performance by using indexing.

Another disadvantage is MongoDB stores denormalised data, meaning the same piece of information can exist in multiple places. This leads to data redundancy and increased storage costs. Moreover, updating redundant data can become complex and error prone, which is a risk if the app frequently updates user data or challenge statuses. For example, deleting a post means that the post IDs must also be deleted from the group collection objects. The impact of storage is negligible as the size of objects is small. The development overhead of manual "garbage collection" is more problematic. However, such problems arise as a result of user data architecture choices rather than an inherent issue with MongoDB itself. This means that it is possible to minimise the impact using appropriate data structures for object properties.

With careful consideration of database architecture such as indexing and data structure of object properties, MongoDB can be a more developer friendly database alternative to SQL while minimise performance overhead. This initial setup can further be reduced when opting for a SaaS platform such as Altas which manages much of the intital configuration.

4.4 Testing

4.4.1 Unit and integration tests

The tests consist of unit, integration, and system testing, with each level covering an increasing range of features. Vladimir Khorikov describes the role of unit testing in software development as "to enable sustainable growth of the software projec" [35]. The use of "sustainable" creates an interesting parallel to the project topic, highlighting the importance of testing with regards to tackling complexity.

For this project, Jest is used as the unit test framework. This lightweight yet powerful tool facilitates test doubles such as mocks and stubs to simulate data for individual units. For the frontend, Jest is combined with React's native testing library to manipulate the UI state and simulate user interactions. In total, 104 unit tests have been written, covering both backend service files and frontend React components. Maintaining high code coverage throughout development increases confidence in the software's quality. With service functions and frontend components being reused multiple times for diverse needs, running unit tests after individual tweaks highlights faults that may raise in unforeseen areas early, preventing their propagation into failures in production.

While unit tests center around the specifications individual components, Integration tests are based on the user stories themselves. By setting user stories as the focal point

of the integration tests, we keep development user centered and focus around user needs. Selenium WebDriver the tool is used to create these UI automation tests. As this is a webview based app, Selenium WebDriver can mimic user interactions and test flows. Unlike the unit tests, these tests reveal how well multiple components work together. We can use these tests as smoke tests to assess production readiness. Having this is essential for continuous deployment, as failures can be identified beforehand, it is the key difference between unit tests which highlights faults/errors and integration/system tests while a adapt at failures. By creating a set of tests to capture faults, errors, and failures, and with strong code coverage we establish a reliable test suit, aiding our development to ensure app quality.

Both Jest and Selenium integrate easily into continuous integration pipelines. This is important if launching into production to sustain a stable branch while working on features or fixing bugs. These tests can be run as part of an automated test suite before merging, providing an indication of quality. The importance of test automation is to achieve continuous delivery. With the code coverage and test tools used, the groundwork is laid for a sustainable CI/CD pipeline that can maintain sustainable growth of the software's complexity.

4.4.2 Exploratory & Usability Tests

4.4.2.1 Testable Hypothesis Providing users with personalised sustainability challenges and immediate feedback, coupled with social interaction and gamification elements, will enhance user engagement, adoption of sustainable practices as well as overall user satisfaction.

4.4.2.2 Target Population The target population consists of demographics are based on the personas of Emma Greenfield, Adnan Yilmaz, and Sarah Williams that encompass the insights obtained from the empathise and design phases. The experiment will involve 12 students.

4.4.2.3 Independent Variables

1. **Customised Challenges** – These are tailored sustainability challenges based on user input about their lifestyle and preferences. This can be shown or hidden.
2. **Immediate Feedback** – This is the real time feedback provided to users about the impact of their actions on sustainability. This can be shown or hidden.
3. **Social Interaction** – This includes features that allow users to join groups, post updates, and engage with others. This can be shown or hidden.
4. **Gamification Elements** – These are elements like leaderboards, badges, and rewards that gamify the user experience. This can be shown or hidden.

4.4.2.4 Dependent Variables

1. **Engagement with the App**: Measured by the frequency and duration of app usage, number of challenges completed, and level of interaction with social features.
2. **Adoption of Sustainable Practices**: Measured by the number of sustainable actions reported, progression in completing challenges, and self reported changes in behavior.
3. **User Satisfaction**: Measured by user feedback collected through surveys and journals, focusing on their experience, perceived impact, and overall satisfaction with the app.

4.4.2.5 Experiment Design The experiment will be a field, between subject design. Each of the 12 students will be randomly assigned to one of four groups, each experiencing a different combination of the independent variables. This approach ensures that we can isolate the effects of each independent variable.

- **Group A**: Customised Challenges + Immediate Feedback
- **Group B**: Customised Challenges + Social Interaction
- **Group C**: Customised Challenges + Gamification Elements
- **Group D**: Control group with no special features (baseline)

Each group will consist of 3 students who share similar attributes with the personas (Emma, Adnan, and Sarah).

4.4.2.6 Threats to Internal Validity

- **Selection Effects**: Random assignment of students to groups to mitigate selection bias.
- **Ordering Effects**: Randomise the order in which features are introduced if applicable.
- **Experimenter Bias**: Use objective measures where possible (e.g., app usage analytics) and ensure journaling instructions are standardised.

4.4.2.7 Threats to External Validity

- **Population Representativeness**: Ensure a diverse sample that reflects the broader student population.
- **Ecological Validity**: Conduct the study in a real world setting to capture authentic user interactions, noting potential external influences through journaling or surveys.

4.4.2.8 Threats to Reliability

- **Consistency in Measurement**: Use standardised tools and methods for data collection, such as built in app analytics and structured surveys.
- **Replication**: Aim to repeat the experiment with a larger sample to verify results.

4.4.2.9 Procedure - Pretest Phase

1. Recruit participants and obtain informed consent.
2. Conduct initial surveys to gather baseline data on engagement, sustainability practices, and satisfaction.
3. Provide training on how to use the app and log their experiences.

4.4.2.10 Procedure - Experiment Phase

1. Assign students to groups and enable the respective features in the app.
2. Monitor app usage, challenge completion, and interactions over a period of 4-6 weeks.
3. Collect qualitative data through participant journals.

4.4.2.11 Procedure - Posttest Phase

1. Conduct follow up surveys to assess changes in engagement, sustainability practices, and satisfaction.
2. Analyze quantitative data from app analytics and qualitative data from journals.

5 Results, Analysis and Evaluation

5.1 Results

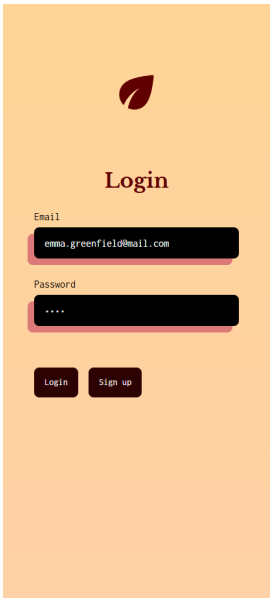

Figure 36: Login screen

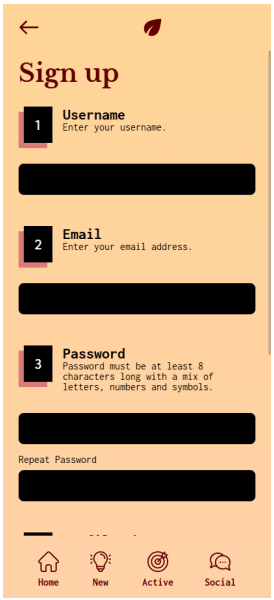

Figure 37: Signup screen

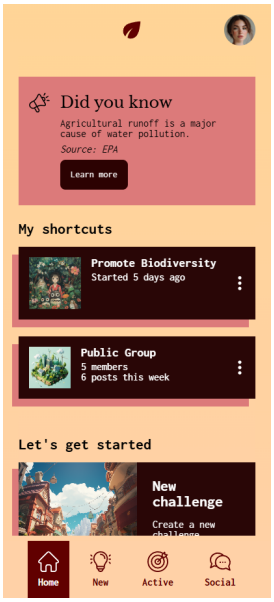

Figure 38: Home screen

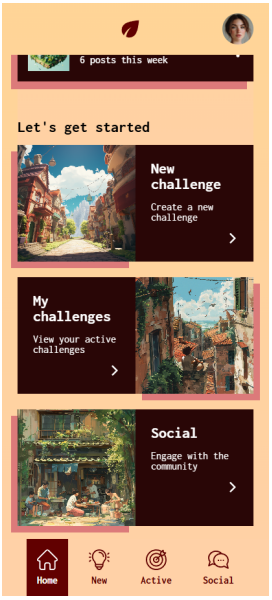

Figure 39: Home screen (continued)

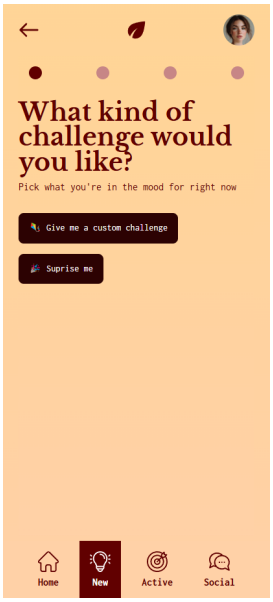

Figure 40: New Challenge Screen 1

Figure 41: New Challenge Screen 2

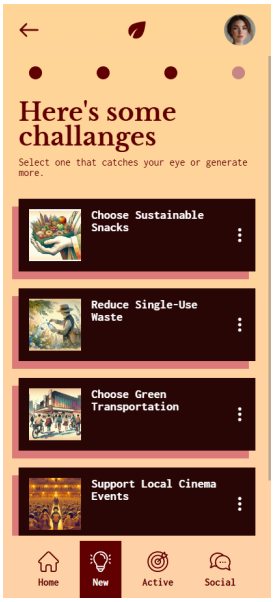

Figure 42: New Challenge Screen 3

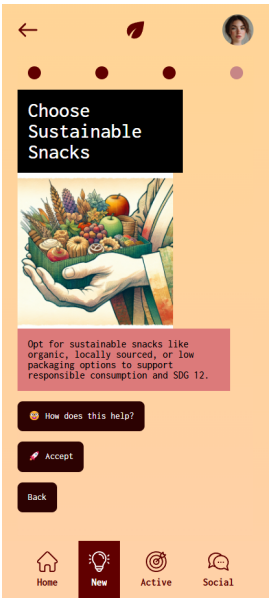

Figure 43: New Challenge Screen 4

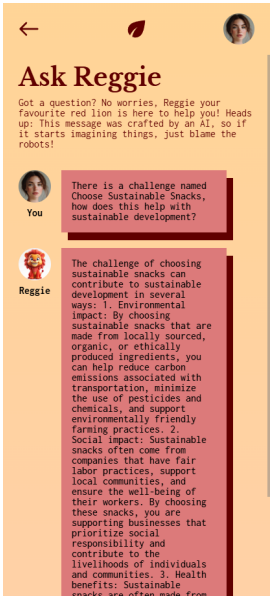

Figure 44: AI Chat overlay 1

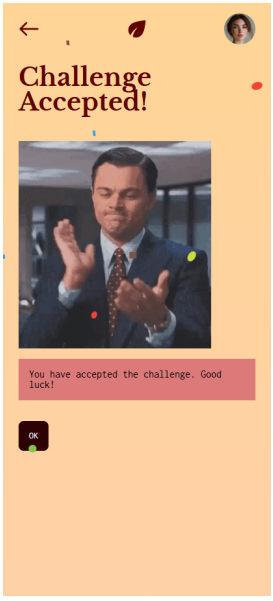

Figure 45: Challenge accepted screen

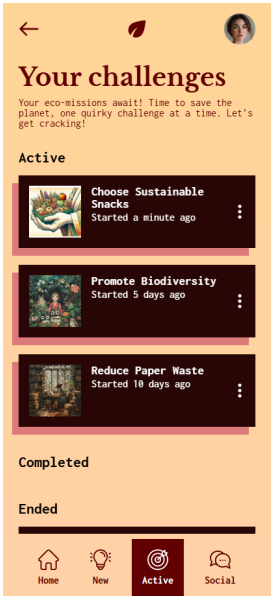

Figure 46: Your challenges screen

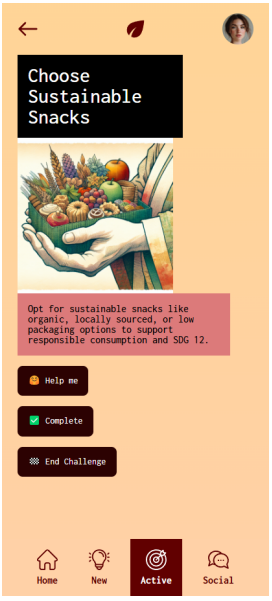

Figure 47: Active challenge screen

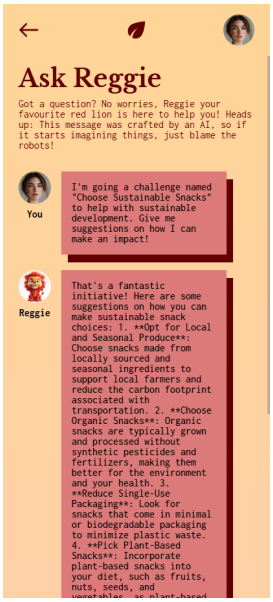

Figure 48: AI Chat overlay 2

Figure 49: Complete challenge screen

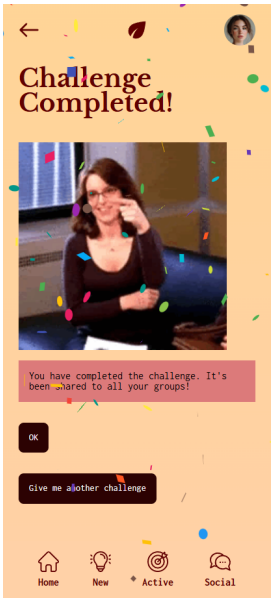

Figure 50: Challenge completed screen

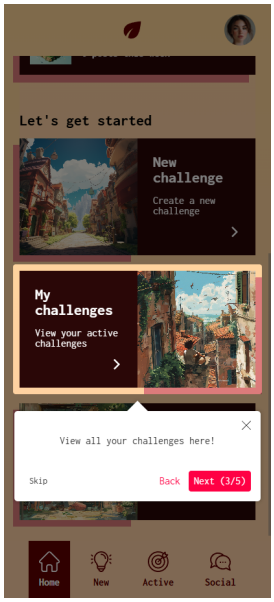

Figure 51: Tutorial overlay

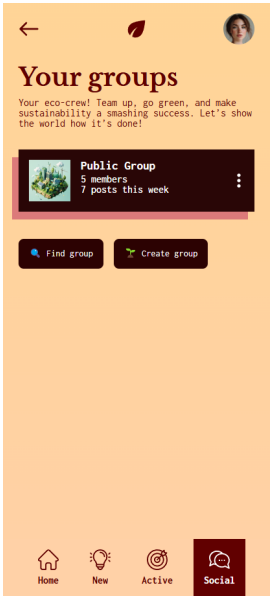

Figure 52: Your groups screen

Figure 53: Find group screen

Figure 54: Group screen

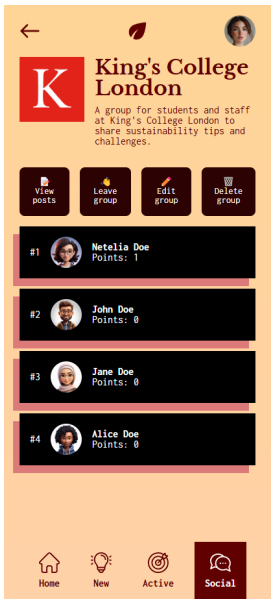

Figure 55: Leaderboard screen

Figure 56: User post screen

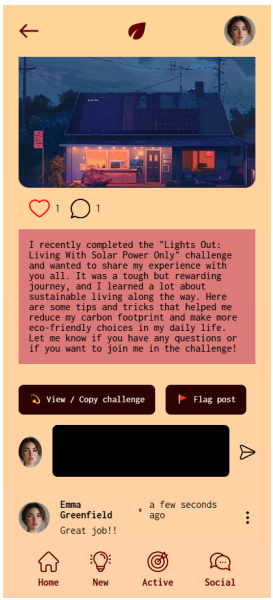

Figure 57: User post screen comments

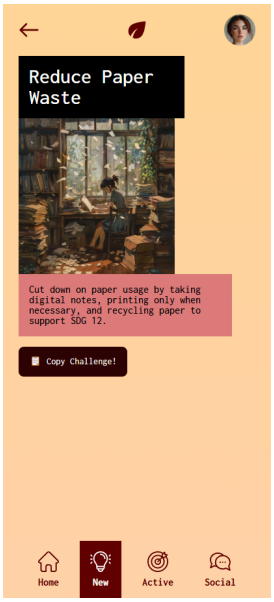

Figure 58: Copy challenge screen

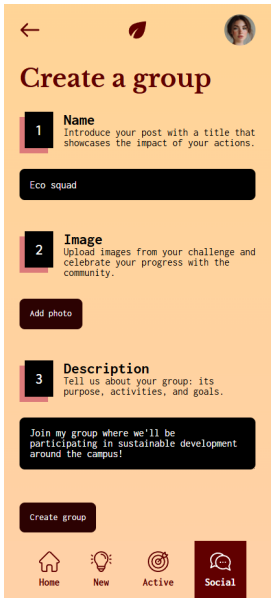

Figure 59: Create group screen

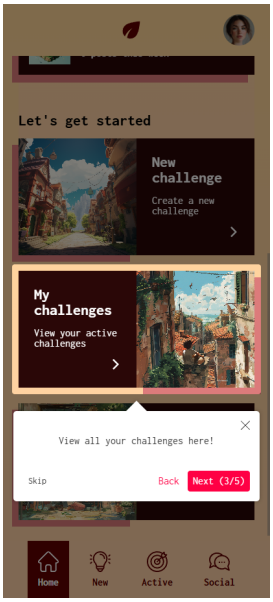

Figure 60: Tutorial overaly

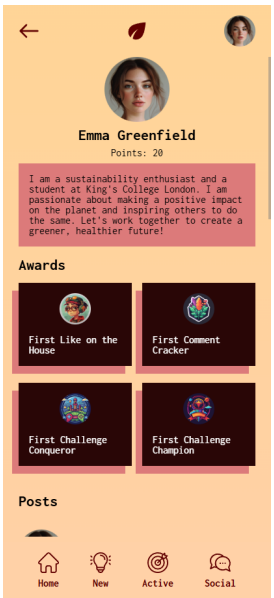

Figure 61: Profile screen

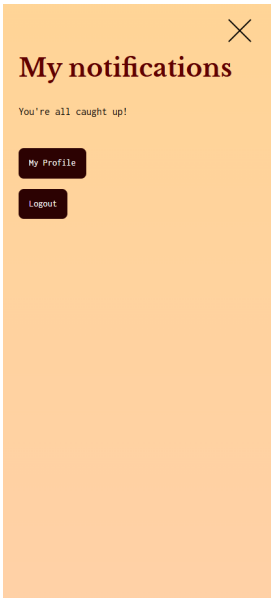

Figure 62: Notifications overlay

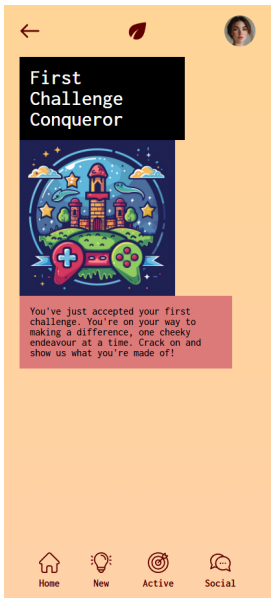

Figure 63: Reward screen

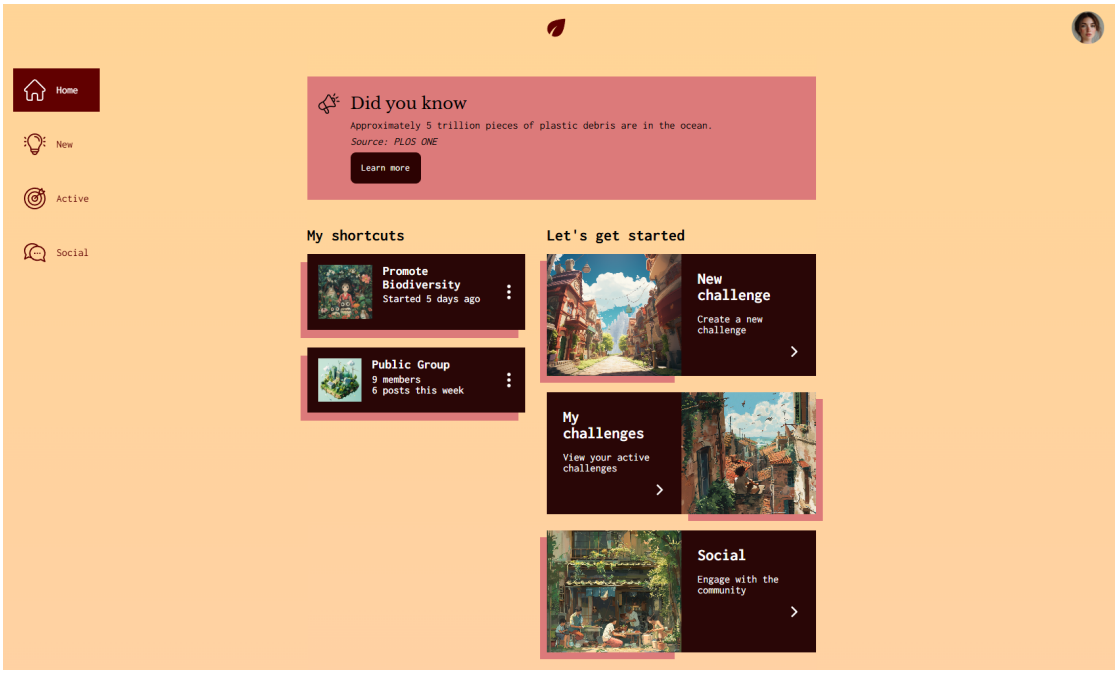

Figure 64: Screen responsiveness example with Home screen

5.2 Evaluation & Analysis of the User Experience

For the analysing the user experience of the app, we will use Norman's design principles as the basis for comparison [36]. These principles guide the creation of intuitive and user friendly interfaces making it a good tool to analyse the user centered approach.

- **Visibility:** Can Emma easily find and understand all the possible actions they can take? A persistent back button and bottom quick navigation leverages the 21:9 aspect ratio common in modern smartphones to increase accessibility of navigation. The colour scheme provides clear contrast, making text and UX elements stand out. Spacing is used effectively to differentiate sections and elements, creating a natural hierarchy that improves navigation.
- **Feedback:** Does the system provide clear, timely, and appropriate responses to the Emma's actions? The app uses progress loading animations for asynchronous actions, such as AI calls, and the toast pattern for feedback of core interactions such as liking posts and accepting challenges keep the user informed.
- **Affordance:** Do the design elements clearly suggest how they can be used? The app effectively uses emojis, to enhance affordance, especially effective among the target audience of students in higher education. Standard icons, such as a back arrow, are also employed to indicate function. Affordance was challenging due to the app's potentially diverse user base, but imagery has proved to be a solution.

- **Mapping:** Is the relationship between controls and their effects intuitive and easy to understand? The app's interactive elements, such as forms for creating groups, incorporate validation rules that guide users to enter valid information. Disabled buttons are used to ensure actions can only be triggered once conditions are satisfied.

- **Consistency:** Are similar actions and elements presented in the same way across the system to avoid confusion? Reusable components, like bottom navigation or specialised challenge items, create familiarity and ease of use across different screens. Using SASS for styling enhances consistency by minimising imports and simplifying the sharing of styles.

- **Constraints:** Are there clear boundaries that prevent user errors or guide them toward correct usage? These can be found in the app's interactive elements, particularly in form elements like group creation or user registration. These screens consist of inputs and buttons with validation rules that guide users to enter valid information. When entering invalid text values, the user is informed of the necessary corrections. By setting a disabled state for buttons, we ensure they can only be triggered once the conditions are satisfied.

- **Error Prevention and Recovery:** Is it easy for users to recognise, diagnose, and recover from errors when they do occur? A help assistant guides users through the app on their first login, helping establish a correct mental model. The persistent back button allows easy navigation back to the previous screen, while the toast pattern with an undo option helps recover from critical actions like deleting a post. Currently, this undo feature is available for post deletions, with potential for future expansion.

5.3 Evaluation of Test Results

The project features a large suite of tests, encompassing both unit and integration tests, with a total of 104 unit tests and 18 UI tests. This extensive coverage is a good indicator of the system's reliability and the effectiveness of the VIPER architecture in promoting testability. The modular nature of VIPER, which naturally decouples logic into distinct layers, has significantly enhanced our ability to maintain visibility and control over the internal system state during testing. By employing the custom hook pattern, we have successfully extracted logic from the UI components, making it easier to test both modules independently. Tools like Jest have proven invaluable, offering comprehensive support for mocks and advanced assertions, which further strengthens our testing framework.

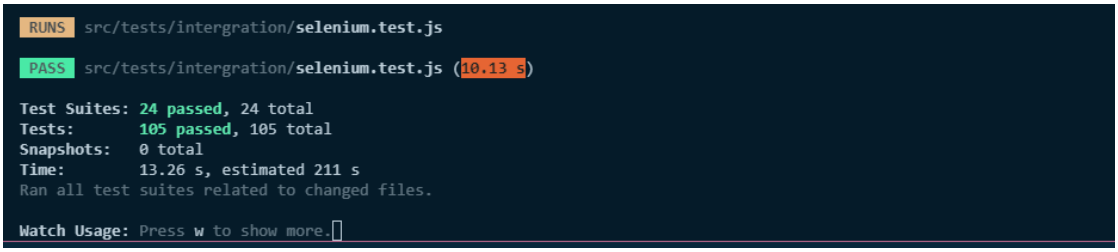

Figure 65: Unit test results

Figure 66: Integration test results based on user stories

However, the challenges of system integration testing differ from those of unit testing, particularly in achieving consistent results across multiple test runs. Unlike unit tests, which can mock data, system tests interact with and modify the actual database state, especially when testing user stories such as ensuring Emma can like other people's posts. To address this, we have implemented a test environment that utilises a separate database from production, ensuring that the data is reset with dummy records before each test run. This approach, along with diligent cleanup practices with tests, significantly improved the consistency of the system tests.

By structuring the system tests around user stories, we have ensured that the core user journeys of the application, are thoroughly validated. This user centered approach has fostered greater empathy for our users' needs, encouraging us to focus more closely on the details that matter most to them. Furthermore, we have identified key smoke tests within our UI testing suit as indicators of production readiness. These tests can serve as a crucial component of a continuous delivery pipeline, providing a metric for assessing the system's readiness for deployment. Similarly, the unit tests can function

as an automated quality gate within the a continuous integration pipeline to ensue that quality of code merged into a source repository is of good standard.

An area for potential improvement lies in the tighter integration of testing within the development workflow, possibly through the adoption of frameworks like Test Driven Development (TDD). While we understand the importance of sustainable development, the decision not to fully implement such frameworks was made to speed up development, given that the software's functionality is not ciritial matters like health or finance. Writing tests still remained a central part of the development process. There is room to enhance testing, particularly in backend service files and frontend hooks. Currently, adding tests around them would only validate the functionality of external libraries like MongoDB and GraphQL rather than our own logic. We could improve by further decoupling and layering the logic into separate modules.

5.4 Evaluation & Analysis against User Stories, Acceptance Criteria and the Problem Statement

The project successfully completes the "must have" user stories that were set out. These serve as the minimum viable product (MVP), meaning the core functionality and essential features of the app have been implemented. These requirements are critical for the app's viability to satisfy the problem statement and proceed to production. Additionally, the majority of the "should have" user stories have also been fulfilled. These add value to the MVP, improving the app's competitiveness. Although there are no direct competitors, this deters others from copying the app. However, the app is not for profit, with the aim of supporting sustainable development goals. Therefore, competitiveness should not be a significant factor in the evaluation.

Similarly, many of the "could have" user stories have also been satisfied. This elevates the app with features such as notifications being implemented, increase confidence in its success if deployed into production.

Finally, the quality of how the acceptance criteria (ACs) have been completed is also of a good standard, often going above and beyond the requirements. For example, the inclusion of loading animations for asynchronous tasks, the use of responsive design, and careful attention to tone in wording all contribute to the user experience. These elements are not captured in the AC but have been added with diligence. The user center approach plays a significant part in this, with better empathy to our user, we far more motivated in delivering the best possible experience.

It is important to note that the user stories themselves are based on assumptions. Although they are data driven, with insights drawn from the literature review, we have yet to assess their impact on actual users and their contribution to sustainable development. For instance, how beneficial are user groups for Emma? Does the AI provide Adnan with relevant explanations? Will Sarah appreciate the profanity filtering features? How much does the tutorial and onboarding experiences help? To validate the legitimacy of these user stories, it is essential to conduct exploratory testing with real users. The usability testing plan will be crucial in addressing these questions and represents an important next step in the app's lifecycle.

Overall, we have set out an array of complex and diverse requirements, grounded in data driven insights from the literature review. These requirements encompass a variety of user stories, including a social media timeline featuring support for likes, comments, and notifications; an AI component that generates personalised sustainable development content and provides chatbot like assistance; user journeys that incorporate help tutorials and onboarding experiences; and competitive elements such as user rewards and leaderboards. The scope of work undertaken is highly ambitious, presenting a vast array of complex challenges.

Given the constraints of time and resources, achieving all of the must have and should have requirements, along with nearly all of the could have requirements, represents a significant accomplishment. This success reflects on the clarity of the requirements through user stories and ACs set out from the beginning, which have proven invaluable throughout the development process.

5.5 Evaluation & Analysis Against Existing Software Platforms

Evaluating the solution against existing apps and games, as highlighted in the literature review, has certain challenges. Although these existing solutions share similarities, they often differ in purpose and intent. For example, while apps like "Cool Choices"integrate gamification elements such as online leaderboards, they focus on specific aspects of sustainable development, in this case, energy reduction. Similarly, other apps like Wasteapp target waste management.

In contrast, this project encompasses all facets of sustainable development through customised challenges. The AI driven personalised features, combined with social media elements, distinguish this app from existing ones by delivering holistic experience. We have also drawn valuable insights from competitors such as "Social Power" which struggled to sustain long term impact and user retention due to a top down approach that lacked the engaging, user centered strategies. To address this, we have employed SHIFT framework into the ideation and specification, promoting more psychological techniques to capture engagement.

Ringorgan employs trivia questions to engage users in learning, which could serve as an effective method for digesting the Sustainable Development Goals (SDGs) within our app. Additionally, Modrdo Shaper, an app that encourages carpooling in Warsaw, Poland, excels at connecting users from the digital world to real world actions through more tangible, real time data. Incorporating a similar technique can further enhance the practical impact of our solution.

In short, there is no existing app or game that encapsulates all facets of sustainable development while providing a platform where users can collaborate and compete with each other in achieving SDGs. This uniqueness gives confidence that the software solution presented by this project is truly novel and innovative, delivering a comprehensive tool that helps people obtain thier SDGs.

5.6 Evaluation & Analysis of the Technical Solution

Many technical challenges have been addressed through the use of a cutting edge technology stack, to bring to life a complex solution which supports sustainable development. The software has been successfully integrated within the VIPER architecture, which divides areas of concern into maintainable modules. Not only has this approach added immediate value improving testability and performance but continue to add future value through simplified maintenance and inherent scalability.

Advanced concepts in React, the most widely used frontend framework, have been implemented, including states, contexts, and custom hooks. These are integrated with various libraries to deliver an interactive experience tailored to the personas of Emma, Adnan, and Sarah. Using GraphQL, we have developed APIs that allow for flexible data retrieval, which has been practically applied to enhance privacy by providing better control over the information transferred across servers. With MongoDB, we have leveraged the benefits of a NoSQL database, such as handling structured data, while optimising performance through indexing to mitigate potential performance disadvantage.

AI has been utilised in a diverse set of features, including personalised content, educational tools, and moderation systems. A comprehensive suite of unit and integration tests has been developed, covering the functional requirements of individual units as well as all implemented user stories. Integration of AWS using S3 buckets has allowed for secure and reliable storage solution.

However, the long term use of WebView remains to be proven. Advances in hardware performance may outpace the growing need for native mobile features. Additionally, the use of OpenAI presents potential privacy and cost issues in the long term. Exploring open source alternatives, such as Meta's Llama LLM, could address these concerns and provide a more sustainable solution.

5.7 Potential extensions

The next phase of the project should involve conducting formal usability testing using the test plan outlined in the implementation section. This will allow us to validate the assumptions made during research and determine whether the user stories address the pain points identified and access the app's impact towards sustainable development.

The use of AI could be further extended to filter harmful content as users create posts, creating further enhance the community for Emma, Adnan, and Sarah. As touched on earlier, exploring the use of open source AI, such as Meta's Llama LLM, is also a key consideration. The costly API structure of OpenAI poses a threat to the app's scalability. Ensuring the app remains free for mass adoption and contribution to the SDGs is crucial. Utilising an open source, self hosted LLM could also alleviate privacy concerns related to users sending personal data. However, further research is needed to assess the performance and capabilities of such AI in the context of SDGs.

In terms of user tracking, exploring anonymous tracking tools like Google Tag Manager and Google Analytics could provide valuable insights into user performance in production. This could enable widespread usability testing using a natural, in the wild

approach, yielding the most authentic results. However, this approach would require overcoming challenges related to privacy management.

Additionally, the project could benefit from utilising the current test suit to create a CI/CD pipeline, automating testing and enhancing development operations. More granular feature level improvements could include implementing an invite system, such as QR codes, allowing Emma's friends or Sarah's students to access the app and join groups with a single click. Furthermore, expanding the use of Reggie to allow custom user prompts and create a more comprehensive chatbot experience. This would require research into the moderation of AI responses and user requests.

5.8 Conclusion

The aim of this project was to create a software solution to help people, particularly students in higher education, contribute to sustainable development. By leveraging generative AI, we can remove many obstacles that currently prevent people from contributing, such as the lack of awareness, feedback on impact, and the need for a more engaging, customised experience. The SHIFT framework has been instrumental in applying marketing ideas to drive user story development, with a strong emphasis on the social aspect. This allows for a platform where users feel part of a broader community, and their achievements are valued.

Rather than focusing on isolated efforts at the institutional level, connecting a global audience toward a single sustainable development goal on a central platform can significantly grow the community by utilising the internet. Gamification concepts integrate well with social features; elements like leaderboards and rewards enable reflection on individual progress while fostering a competitive and engaging culture. However, the use of AI introduces concerns about user data privacy, especially when using external services. AI driven moderation also raises issues around censorship and questions about the model's training and potential biases.

VIPER has proven to be an effective architecture for building software with a focus on complex user interactions. Its separation of concerns breaks down logic into well defined layers, increasing overall maintainability and testability. While GraphQL typically has lower API response times compared to REST APIs, its robust and explicit nature allows for more predictable code. With good architectural design and developer patterns in place, REST APIs remain a competitive option for building the backend. React has been an excellent framework for creating dynamic UIs, with its impressive community providing solutions for a wide range of issues, including routing, testing, and bespoke UI needs such as animation and validation.

NoSQL databases like MongoDB offer many benefits, including the ability to store structured data and seamless integration with a JavaScript based technology stack. However, to achieve performance that remains competitive with SQL databases, careful consideration is required when structuring collections, such as indexing. Overall, the MERN stack dispute created as a web framework, has proven to be effective for building mobile focused software using Webview.

A user centered design approach facilitates excellent empathy with the user during

development. By integrating user needs and motivations into the user stories, we can focus the requirements around the user, ensuring that the software delivers an experience that is both inclusive and user friendly. While a user centered approach requires additional investment in resources during the design stage, particularly in gaining access to a diverse range of user needs, this investment is crucial. Without proper representation, there is a risk of producing biased results that fail to address actual pain points. However, for user facing software, a user centered approach has proved to be invaluable as it consistently highlights pain points throughout all stages of development, promoting decision making to technical problems that genuinely benefit the user.

6 Legal, Social, Ethical, and Professional Issues

6.1 Professional Standards

Throughout the project, adherence to relevant professional standards has been a priority. The software development process has been guided by the British Computer Society (BCS) Code of Conduct and Code of Good Practice, ensuring that the project meets the ethical and professional standards expected in the field of software engineering. These standards highlight the importance of public safety, privacy, and well being. Additionally they provide guidelines on responsible professional behavior, including correct use of third party materials and proper referencing of external work.

In addition, the project has been mindful of intellectual property laws and licensing requirements. All third party libraries and tools used in the development process have been carefully reviewed to ensure compliance with their respective licenses. This diligence aims to uphold the integrity of the project and rights of original content creators. Doing so we aim to prevents legal issues related around misuse of intellectual property.

6.2 Social and Ethical Implications

The software developed in this project has significant social and ethical implications, especially in the context of sustainable development promotion. By encouraging users to participate in sustainability challenges, the software aims to create behaviour that is environmentally responsible. However, this raises ethical concerns related to user motivation and behavior change. It is important that the software avoids manipulative tactics and instead focuses on empowering users to make voluntary and informed decisions to contribute towards sustainability.

Furthermore, the use of competition as a motivational tool must be carefully monitored to ensure that it creates a healthy and positive engagements rather than causing excessive competition, stress and poor mental health. The software should aim to create an inclusive environment that encourages participation from all users, regardless of their starting level of knowledge or commitment to sustainability.

Use of AI also raises issue around the integrity of the information provided. Hallucinations are a common phenomena within LLM with puts into question the validity

of information directed at users in shaping their understanding of SD. Concerns over spreading misinformation or biased agenda must be looked into.

6.3 Impact on Public Well being and Sustainability

The impact of this project extends beyond individual users to broader societal and environmental contexts. By promoting sustainable development practices, the software has the potential add positive value towards public well being and global sustainability. Through SDGs we encourage users to adopt behaviors that reduce their ecological footprint, conserve resources, and support other people around thier community.

However, we must also understand the potential economic and commercial impact our implementation. While the primary goal is to promote sustainability, the software needs to take into account the economic reality faced by all users. For example, It is important that the challenges and recommendations provided by the software are practical and accessible to users from diverse economic backgrounds.

In conclusion, this project is committed to upholding highest standards of professional conduct while addressing the complex social, ethical, and environmental challenges linked with promoting sustainable development through software. By considering the broader impact of the software to the community, we aim to make a meaningful contribution not only in the field of software engineering, but also the global efforts towards sustainability.

References

[1] C. Allen, G. Metternicht, and T. Wiedmann, "Initial progress in implementing the sustainable development goals (sdgs): A review of evidence from countries," *Sustainability science*, vol. 13, pp. 1453–1467, 2018.

[2] A. V. Agbedahin, "Sustainable development, education for sustainable development, and the 2030 agenda for sustainable development: Emergence, efficacy, eminence, and future," *Sustainable Development*, vol. 27, no. 4, pp. 669–680, 2019.

[3] D. Osborn, A. Cutter, and F. Ullah, "Universal sustainable development goals," *Understanding the transformational challenge for developed countries*, vol. 2, no. 1, pp. 1–25, 2015.

[4] W. Katy, "Seed youth listening project report 2021-23," *American psychologist*, vol. 1, no. 1, p. 38, 2023.

[5] W. Leal Filho, Y.-C. J. Wu, L. L. Brandli, L. V. Avila, U. M. Azeiteiro, S. Caeiro, and L. R. d. R. G. Madruga, "Identifying and overcoming obstacles to the implementation of sustainable development at universities," *Journal of Integrative Environmental Sciences*, vol. 14, no. 1, pp. 93–108, 2017.

[6] L. Velazquez, N. Munguia, and M. Sanchez, "Deterring sustainability in higher education institutions: An appraisal of the factors which influence sustainability in higher education institutions," *International Journal of Sustainability in Higher Education*, vol. 6, no. 4, pp. 383–391, 2005.

[7] J. C. Stephens, M. E. Hernandez, M. Román, A. C. Graham, and R. W. Scholz, "Higher education as a change agent for sustainability in different cultures and contexts," *International journal of sustainability in higher education*, vol. 9, no. 3, pp. 317–338, 2008.

[8] K. Shephard, "Higher education's role in'education for sustainability'," *TheAustralian Universities' Review*, vol. 52, no. 1, pp. 13–22, 2010.

[9] M. Despeisse, "Teaching sustainability leadership in manufacturing: A reflection on the educational benefits of the board game factory heroes," *Procedia CIRP*, vol. 69, pp. 621–626, 2018. 25th CIRP Life Cycle Engineering (LCE) Conference, 30 April – 2 May 2018, Copenhagen, Denmark.

[10] J. N. Meya and K. Eisenack, "Effectiveness of gaming for communicating and teaching climate change," *Climatic change*, vol. 149, no. 3, pp. 319–333, 2018.

[11] A. L. Beck, S. Chitalia, and V. Rai, "Not so gameful: A critical review of gamification in mobile energy applications," *Energy Research Social Science*, vol. 51, pp. 32–39, 2019.

[12] A. Lieberoth, N. Holm Jensen, and T. Bredahl, "Selective psychological effects of nudging, gamification and rational information in converting commuters from cars to buses: A controlled field experiment," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 55, pp. 246–261, 2018.

[13] R. Gifford, "The dragons of inaction: psychological barriers that limit climate change mitigation and adaptation.," *American psychologist*, vol. 66, no. 4, p. 290, 2011.

[14] R. B. Cialdini, L. J. Demaine, B. J. Sagarin, D. W. Barrett, K. Rhoads, and P. L. Winter, "Managing social norms for persuasive impact," *Social influence*, vol. 1, no. 1, pp. 3–15, 2006.

[15] T. Green and J. Peloza, "Finding the right shade of green: The effect of advertising appeal type on environmentally friendly consumption," *Journal of advertising*, vol. 43, no. 2, pp. 128–141, 2014.

[16] H. Tajfel and J. C. Turner, "The social identity theory of intergroup behavior," in *Political psychology*, pp. 276–293, Psychology Press, 2004.

[17] T. Kurz, B. Gardner, B. Verplanken, and C. Abraham, "Habitual behaviors or patterns of practice? explaining and changing repetitive climate-relevant actions," *Wiley Interdisciplinary Reviews: Climate Change*, vol. 6, no. 1, pp. 113–128, 2015.

[18] D. Dunning, "Self-image motives and consumer behavior: How sacrosanct self-beliefs sway preferences in the marketplace," *Journal of consumer psychology*, vol. 17, no. 4, pp. 237–249, 2007.

[19] E. Van der Werff, L. Steg, and K. Keizer, "I am what i am, by looking past the present: The influence of biospheric values and past behavior on environmental self-identity," *Environment and behavior*, vol. 46, no. 5, pp. 626–657, 2014.

[20] A. Bandura, "Self-efficacy: toward a unifying theory of behavioral change.," *Psychological review*, vol. 84, no. 2, p. 191, 1977.

[21] V. Corral-Verdugo, M. Bonnes, C. Tapia-Fonllem, B. Fraijo-Sing, M. Frías-Armenta, and G. Carrus, "Correlates of pro-sustainability orientation: The affinity towards diversity," *Journal of environmental psychology*, vol. 29, no. 1, pp. 34–43, 2009.

[22] S. Banerjee, C. S. Gulas, and E. Iyer, "Shades of green: A multidimensional analysis of environmental advertising," *Journal of advertising*, vol. 24, no. 2, pp. 21–31, 1995.

[23] R. W. Reczek, R. Trudel, and K. White, "Focusing on the forest or the trees: How abstract versus concrete construal level predicts responses to eco-friendly products," *Journal of Environmental Psychology*, vol. 57, pp. 87–98, 2018.

[24] G. D. Greenwade, "The comprehensive tex archive network (ctan)," *TUGBoat*, vol. 14, no. 3, pp. 342–351, 1993.

[25] J. Pruitt and J. Grudin, "Personas: practice and theory," in *Proceedings of the 2003 conference on Designing for user experiences*, pp. 1–15, 2003.

[26] J. Arnowitz, M. Arent, and N. Berger, *Effective prototyping for software makers*. Elsevier, 2010.

[27] M. Cohn, *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.

[28] R. Razzouk and V. Shute, "What is design thinking and why is it important?," *Review of educational research*, vol. 82, no. 3, pp. 330–348, 2012.

[29] J. A. Khan, I. U. Rehman, Y. H. Khan, I. J. Khan, and S. Rashid, "Comparison of requirement prioritization techniques to find best prioritization technique," *International Journal of Modern Education and Computer Science*, vol. 7, no. 11, p. 53, 2015.

[30] D. Kulak and E. Guiney, *Use cases: requirements in context*. Addison-Wesley, 2012.

[31] G. J. Myers, C. Sandler, and T. Badgett, *The art of software testing*. John Wiley & Sons, 2011.

[32] E. Saks, "Javascript frameworks: Angular vs react vs vue.," *International Journal of Modern Education and Computer Science*, 2019.

[33] G. Brito and M. T. Valente, "Rest vs graphql: A controlled experiment," in *2020 IEEE International Conference on Software Architecture (ICSA)*, pp. 81–91, 2020.

[34] A. Makris, K. Tserpes, G. Spiliopoulos, D. Zissis, and D. Anagnostopoulos, "Mongodb vs postgresql: A comparative study on performance aspects," *GeoInformatica*, vol. 25, pp. 243–268, 2021.

[35] V. Khorikov, *Unit Testing Principles, Practices, and Patterns*. Simon and Schuster, 2020.

[36] W. Lidwell, K. Holden, and J. Butler, *Universal principles of design, revised and updated: 125 ways to enhance usability, influence perception, increase appeal, make better design decisions, and teach through design*. Rockport Pub, 2010.

A Appendix

A.1 Setting up the app locally

As this is a web app, we can setup the project to be viewed through the browser without installing an emulator such as Android studio.

A.1.1 Setting up Node

1. Install Node.js v16.16.0.
2. Install npm 9.6.2.
3. open a command prompt at the root source folder.
4. run “npm install”.

A.1.2 Setting up database

1. Install MongoDB Community Server 7.0.x.
2. Launch MongoDB Community Server.
3. Run the database setup script at “server/utils/database_setup.js”.

A.1.3 Setting up the server

1. open a command prompt at the root source folder
2. run “node ./server/server-graphql.js”

A.1.4 Setting up the frontend

For additional help, refer to the README.md file at root directory.

1. open a command prompt at the root source folder.
2. run “npm run start”.

A.1.5 Setting up the tests

1. open a command prompt at the root source folder.
2. run “npm run test” .

A.1.6 Viewing as mobile (through browser emulation)

1. launch the app.
2. open developer tools from you browser (e.g. Ctrl + Shift + C in Chrome for Windows).
3. select mobile view from your brower (e.g. toggle device emulation in Chrome).