Problem Analysis

We need to create a roadmap to address how we will review user feedback, maintain the existing system, and design new features, across a timespan of 6 months with 3 engineers and one designer.

This will involve a review of the current user experience and needs as well as making key decisions for the technical future of the project and a presentation to effectively communicate to stakeholders.

Background to the problem:

RunWize, a promising new entrant in the fitness app market, has been operational for just over a year with a bold ambition to become the leading fitness app in the UK. The app, available on both iOS and Android platforms, including smartwatches, aims to empower runners, transform lives, and foster a vibrant community centred on health, resilience, and achievement.

Despite its mission, RunWize has encountered a significant challenge in recent months. Over the past three months, there has been a noticeable decline in the number of monthly app users. Compounding this issue, the app has experienced a churn rate of 10% among its paid subscribers since the beginning of the year.

To address these challenges and realign with its growth ambitions, RunWize has set a critical business objective: to enhance user engagement and boost subscription growth. The key results targeted to achieve this objective include:

- Achieving a 15% increase in Monthly Average Users (MAU) within the next six months.
- Increasing the paid subscriber count by 20% within the same period.

To prioritize and tackle these issues effectively, RunWize must implement a severity scale, categorizing the urgency and impact of various strategies and actions into Low, Medium, High, and Critical levels. This approach will ensure that the most pressing issues are addressed promptly, thereby improving overall user engagement and subscription growth.

Client feedback:

User feedback is crucial for the continuous improvement and success of any product or service. For an app like RunWize, user feedback provides direct insights into the user experience, helping to identify pain points, areas for enhancement, and new feature opportunities. Listening to users ensures that the app evolves to meet their needs and expectations, fostering loyalty and reducing churn. It also allows the company to stay competitive by quickly adapting to trends and user demands. By acting on feedback, RunWize can enhance user satisfaction, increase engagement, and achieve its business objectives more effectively.

1. Weekly Distance Targets Flexibility: Users, especially beginners, find the current distance targets discouraging and seek more customizable goals that

align with their fitness levels. Implementing flexible goal-setting options could make the app more inclusive and encouraging for new runners.

2. Guidance on Metrics Interpretation: Beginners are unsure how to interpret metrics like stride length and cadence. Providing educational content or interactive guides on these metrics can enhance the user experience and make the app more user-friendly for all fitness levels.
3. Notification Issues: The reminders feature is unreliable, not always sending notifications on the correct day. Ensuring the notifications work consistently and accurately is essential to maintain user engagement and trust.
4. Post-Run Analysis: Users desire more detailed post-run analysis and actionable feedback. Enhancing this feature can help users track their progress more effectively and stay motivated by understanding their performance better.
5. Advanced Training Features: There is a demand for advanced training features such as interval training programs and customizable workout plans. Adding these features would cater to more serious runners and those training for specific events, making the app more versatile.
6. Heart Rate Insights: Users appreciate heart rate tracking but want advanced insights and the ability to create heart rate-based training zones. Offering these advanced analytics can help users optimize their training and achieve better results.
7. Diverse Goal Setting: While weekly distance targets are appreciated, users also want to set goals related to pace improvement or the number of runs per week. Introducing various goal-setting options would cater to different user preferences and training styles.
8. Enhanced Social Features: The app's current social sharing features are seen as basic. Users are interested in more community interaction, such as joining virtual running clubs or participating in challenges. Enhancing social features could foster a stronger sense of community and increase user engagement.
9. Personalized Reminders: The reminder system lacks personalization. Users would benefit from reminders that adapt to their individual running habits and rest days, making the app more supportive and aligned with their routines.
10. Privacy Concerns: Users are concerned about the safety of sharing their running routes publicly. Implementing more robust privacy settings and safety features would address these concerns, making users feel more secure when using the app.

By addressing these feedback points, RunWize can significantly improve its user experience, cater to a broader audience, and achieve its key business objectives of enhancing user engagement and growing its subscriber base.

Competitor research:

Competitor research is vital for staying competitive in the fitness app market. By analysing similar products, such as Fitbit and Apple Watch, RunWize can identify industry standards and popular features that users expect. These competitors often offer advanced metrics tracking, personalized coaching, detailed post-workout analysis, customizable training plans, and robust social features. Incorporating these elements into RunWize can enhance its appeal, ensuring it meets and exceeds user expectations. This strategic approach helps RunWize not only match but potentially surpass its competitors, driving higher user engagement and satisfaction: Apple Watch, with a significant share of the smartwatch market, provides detailed post-workout analysis, customizable training plans, and advanced health insights like ECG monitoring and blood oxygen levels.

Expected and Desired Features Based on Competitor Analysis:

- **Advanced Metrics Tracking**: Fitbit and Apple Watch both offer extensive health metrics, such as heart rate variability, VO2 max, and sleep quality. Incorporating similar in-depth tracking in RunWize can help users gain a comprehensive understanding of their fitness levels.
- **Personalized Coaching and Training Plans**: These competitors provide tailored workout recommendations and training plans based on user data. RunWize can enhance user experience by offering personalized coaching and adaptable training schedules to meet individual fitness goals.
- **Detailed Post-Workout Analysis**: Apple Watch users benefit from in-depth post-workout reports, including metrics on pace, distance, and recovery. Adding detailed analysis and actionable feedback on RunWize can help users improve their performance effectively.
- **Customizable Training Features**: Fitbit and Apple Watch allow users to customize their workout routines and set specific goals, such as interval training or race preparation. Implementing similar customizable options in RunWize can cater to a broader range of fitness enthusiasts, from beginners to advanced athletes.
- **Robust Social and Community Features**: Fitbit has a strong community aspect with features like challenges and social sharing, engaging users through competition and support. Enhancing RunWize's social features to include virtual running clubs, challenges, and improved social sharing can foster a vibrant user community and boost engagement.

By integrating these advanced features and leveraging the successful use cases of leading competitors, RunWize can significantly enhance its appeal, ensuring it meets and exceeds user expectations, driving higher user engagement and satisfaction.

Available project resources:

The RunWize project is supported by a dedicated team comprising three engineers and one designer, all committed to enhancing the app's functionality and user experience. Leveraging the power of the internet and a suite of project management tools, the team can efficiently organize and allocate their efforts. Tools such as

precedence tables, Gantt charts, and resource histograms will be instrumental in planning and tracking the project's progress. Precedence tables will help in identifying the sequence of tasks and their dependencies, ensuring a streamlined workflow. Gantt charts will provide a visual timeline of the project, allowing the team to monitor milestones and deadlines effectively. Resource histograms will aid in balancing workloads and ensuring that each team member's skills are utilized optimally. This structured approach will enable the team to manage their time and resources efficiently, ensuring that the project stays on track and meets its objectives.

Examples of tools that we can use to manage these features include https://www.madebyevan.com/fsm/ and Microsoft excel and access

Feedback breakdown:

Feedback	Severity	Type
Reminders on the right day	High	Bug-fix
Better privacy and safety	High	Feature
Ability to set own goals	Medium	Feature
Better UI/Help menu to interpret data	Medium	Feature
More feedback post-run	Medium	Feature
Setting other targets	Medium	Feature
Personalised reminders	Medium	Feature
Missing advanced training features	Low	Feature
Heart-rate insights	Low	Feature
Community interaction	Low	Feature

Feasible solutions:

To address the challenges and achieve the business objectives of RunWize, several feasible solutions can be considered. Each solution focuses on different aspects of the app's development and enhancement.

Solution 1: Multiple Streams (Subscriber Stream and Standard Stream)

Overview: This solution involves creating two parallel development streams: one for paid subscribers and one for standard users. This approach allows the team to cater to the specific needs of both user groups simultaneously.

Key Actions:

- Subscriber Stream:
    - Implement advanced features such as customizable training plans, heart rate-based training zones, and detailed post-run analysis.
    - Enhance social features, including virtual running clubs and exclusive challenges for subscribers.

    - Develop personalized coaching and adaptive goal setting based on user data.
- Standard Stream:
    - Improve basic features like goal flexibility, educational content on metrics, and reliable notification systems.
    - Introduce a tiered system where standard users can gradually unlock more features as they engage more with the app.

Resources Allocation:

- Engineers: Split into two groups, each focusing on one stream.
- Designer: Work across both streams, ensuring a cohesive user experience and consistent design language.

Expected Outcomes:

- Increased user engagement and satisfaction in both user groups.
- Higher conversion rates from standard to paid subscribers.

Solution 2: Bug-Fixing Priority

Overview: Prioritizing bug-fixing can significantly improve the user experience by ensuring the app runs smoothly and reliably. This solution focuses on addressing the most critical issues before implementing new features.

Key Actions:

- Conduct a comprehensive audit of the app to identify and prioritize bugs.
- Allocate a significant portion of the team's time to fix the most impactful bugs, such as notification issues and inaccuracies in metrics.
- Implement a robust testing phase to prevent future bugs and ensure new features do not introduce new issues.

Resources Allocation:

- Engineers: All engineers initially focus on bug-fixing, gradually shifting to new feature development as bugs are resolved.
- Designer: Assist in identifying usability issues related to bugs and ensure that fixes align with the overall user experience.

Expected Outcomes:

- Improved app stability and user trust.
- Enhanced user satisfaction and retention due to a smoother, more reliable experience.

Solution 3: Integrating the Designer into the Problem-Solving Process

Overview: The designer plays a crucial role in enhancing the user experience and interface, ensuring that all changes are visually appealing and user-friendly. This solution emphasizes the integration of the designer into the entire development process.

Key Actions:

- The designer collaborates closely with engineers from the start to ensure that all new features and bug fixes maintain a high standard of usability and aesthetics.
- Conduct user research and usability testing to gather insights and validate design choices.
- Develop a cohesive design system that allows for consistent and efficient updates across both the subscriber and standard streams.

Resources Allocation:

- Designer: Actively involved in all stages of development, from initial concept to final implementation.
- Engineers: Work alongside the designer to implement visually cohesive and user-friendly features.

Expected Outcomes:

- A more intuitive and engaging user interface.
- Higher user satisfaction due to a seamless and visually appealing app experience.
- Better alignment between functional updates and user experience enhancements.

Chosen solution:

The chosen solution for RunWize blends the prioritization of bug-fixing (Solution 2) for the first three months, followed by the implementation of multiple development streams for subscribers and standard users (Solution 1). This approach ensures a stable and reliable app before introducing advanced features and tailored experiences.

Phase 1: Bug-Fixing Priority (Months 1-3)

Objective: Improve app stability and user trust by addressing critical bugs and usability issues.

Key Actions:

1. Comprehensive Bug Audit (Month 1):

- Conduct a thorough audit to identify and categorize all existing bugs.
- Prioritize bugs based on their impact on user experience and functionality.

2. Bug Resolution (Months 1-3):
    - Allocate all engineers to fix the most critical bugs first, followed by less severe issues.
    - Implement rigorous testing to ensure bugs are resolved without introducing new issues.
    - Schedule weekly reviews to track progress and reprioritize if necessary.
3. User Feedback Integration (Ongoing):
    - Continuously gather and analyse user feedback to identify any new or persistent issues.
    - Adjust bug-fixing priorities based on user reports and feedback.
4. Review and Assessment (End of Month 1):
    - Evaluate the progress of bug-fixing efforts.
    - Adjust the plan if necessary, ensuring that critical bugs are resolved promptly.

Resources Allocation:

- Engineers: All engineers focused on bug-fixing.
- Designer: Assist in identifying and resolving usability issues related to bugs.

Expected Outcomes:

- Significant reduction in app crashes, glitches, and other technical issues.
- Enhanced user trust and satisfaction due to a smoother, more reliable app experience.

Phase 2: Implementation of Multiple Streams (Months 4-6)

Objective: Enhance user engagement and subscription growth by introducing tailored features for both subscribers and standard users.

Key Actions:

1. Planning and Design (Month 3):
    - Define the scope and specific features for the subscriber and standard streams.
    - The designer develops a cohesive design plan that aligns with new features and existing app aesthetics.

2. Development of Subscriber Stream (Months 4-6):
    - Implement advanced features such as customizable training plans, heart rate-based training zones, and detailed post-run analysis.
    - Enhance social features, including virtual running clubs and exclusive challenges.
3. Development of Standard Stream (Months 4-6):
    - Improve basic features like goal flexibility, educational content on metrics, and reliable notification systems.
    - Introduce a tiered system where standard users can gradually unlock more features as they engage more with the app.
4. User Testing and Feedback (Ongoing):
    - Conduct user testing sessions to gather feedback on new features.
    - Iterate on the design and functionality based on user input.
5. Marketing and Launch (Month 6):
    - Develop a marketing strategy to promote new features.
    - Roll out the new features to users with a clear communication plan highlighting the improvements.

Resources Allocation:

- Engineers: Split into two groups, each focusing on one stream.
- Designer: Work across both streams, ensuring a cohesive user experience and consistent design language.

Expected Outcomes:

- Increased user engagement and satisfaction in both user groups.
- Higher conversion rates from standard to paid subscribers.
- Enhanced app functionality and user experience**,** aligning with RunWize's mission and business objectives.

Documented Design

Performance metrics:

To track the progress and effectiveness of the RunWize project, we need to identify key performance metrics that align with the project objectives and phases. These

metrics will help in evaluating the success of bug-fixing efforts and the implementation of new features, as well as in assessing overall user engagement and satisfaction.

Phase 1: Bug-Fixing Priority (Months 1-3)

1. Bug Resolution Rate

- Description: The number of bugs identified and fixed over a specific period.
- Measurement Method: Track the number of bugs reported, prioritized, and resolved each week. Use a bug tracking system (e.g., Jira, Bugzilla) to log and monitor progress.
- Target: Resolve 80% of high-priority bugs and 50% of medium-priority bugs within the first two months.

2. App Stability

- Description: The frequency of app crashes and major glitches reported by users.
- Measurement Method: Monitor crash reports and stability metrics through analytics tools (e.g., Firebase Crashlytics, Sentry). Measure the reduction in crash rate.
- Target: Achieve a 50% reduction in app crashes by the end of Month 3.

3. User Feedback on Bugs

- Description: The volume and nature of user feedback related to bugs and usability issues.
- Measurement Method: Collect and analyse user feedback through in-app surveys, app store reviews, and customer support interactions.
- Target: Decrease the number of negative feedback reports related to bugs by 40% within three months.

Phase 2: Implementation of Multiple Streams (Months 4-6)

1. Monthly Active Users (MAU)

- Description: The number of unique users who engage with the app at least once a month.
- Measurement Method: Track user activity through analytics tools (e.g., Google Analytics, Mixpanel). Compare MAU before and after feature implementation.
- Target: Achieve a 15% increase in MAU within six months.

2. Paid Subscriber Growth

- Description: The number of new paid subscribers acquired each month.

- Measurement Method: Monitor subscription data through the app's backend and analytics platforms. Measure the rate of new subscriptions and retention.
- Target: Increase the paid subscriber count by 20% within six months.

3. Feature Usage Rate

- Description: The frequency of use of newly implemented features (e.g., customizable training plans, heart rate-based training zones).
- Measurement Method: Track feature-specific engagement metrics using in-app analytics. Measure how often and how extensively users interact with new features.
- Target: Ensure that at least 30% of active users engage with the new features within three months of their release.

4. User Satisfaction

- Description: The overall satisfaction level of users with the new features and app improvements.
- Measurement Method: Conduct user satisfaction surveys and analyze app store ratings and reviews. Measure changes in average rating and satisfaction scores.
- Target: Achieve an average user satisfaction score of 4.5 out of 5 for new features.

5. Feature Adoption Rate

- Description: The percentage of users who adopt and regularly use new features.
- Measurement Method: Analyse usage patterns and adoption rates through analytics tools. Measure the proportion of users who try new features and continue using them over time.
- Target: Attain a 50% adoption rate for key new features within three months of their introduction.

Technical methodologies and stack:

The technical methodologies chosen for a project significantly impact its development and deployment. For instance, selecting Agile methodologies like Scrum or Kanban promotes iterative development, enabling quick adaptation to changing requirements and early delivery of valuable features. Agile fosters collaboration and transparency among team members and stakeholders, enhancing project flexibility and responsiveness. Conversely, Waterfall methodology, characterized by sequential phases from requirements to deployment, provides a structured approach suitable for projects with stable requirements and defined outcomes. Each methodology has its strengths: Agile supports dynamic projects with

evolving needs, while Waterfall ensures thorough planning and adherence to initial scope.

Agile Methodology Benefits:

1. **Adaptability:**
    - **Changing Requirements:** Agile is well-suited for projects where requirements might evolve over time. Given that we plan to start with bug-fixing and then move on to feature development, Agile allows for adjustments based on user feedback and ongoing testing.
2. **Incremental Progress:**
    - **Frequent Deliverables:** Agile enables incremental development, delivering small but functional parts of the project regularly. This aligns well with our goal of improving user engagement and subscription growth through continuous enhancements.
3. **Collaborative Approach:**
    - **Team Collaboration:** Agile promotes collaboration between engineers, designers, and stakeholders. Regular meetings and updates ensure that everyone is aligned with the project goals and can address issues promptly.
4. **Flexibility in Resource Allocation:**
    - **Dynamic Resource Management:** With Agile, we can dynamically allocate our three engineers and one designer to different tasks based on current priorities, ensuring efficient use of your team's skills and time.

Why Not Waterfall?

- **Rigid Structure:**
    - Waterfall’s linear and sequential approach would be less effective given the need for ongoing adjustments and iterative development.
- **Delayed Feedback:**
    - In Waterfall, feedback is typically gathered at the end of the project phases, which could lead to late identification of issues, whereas Agile promotes continuous feedback and iteration.

Activity Precedence:

To manage the project effectively, we will split the tasks into manageable chunks that take between 1-4 weeks to complete. This breakdown will facilitate critical path analysis and ensure that tasks are started and progressed within the project timeframe.

Phase 1: Bug-Fixing Priority (Months 1-3)

Month 1: Comprehensive Bug Audit

- Task 1.1: Identify and categorize all existing bugs (1 week)
- Task 1.2: Prioritize bugs based on impact (1 week)
- Task 1.3: Develop a bug resolution plan (1 week)
- Task 1.4: Initial testing and verification of bug reports (1 week)

Months 2-3: Bug Resolution and Continuous Feedback

- Task 2.1: Fix high-priority bugs (2 weeks)
- Task 2.2: Conduct thorough testing of fixed bugs (1 week)
- Task 2.3: Fix medium-priority bugs (2 weeks)
- Task 2.4: Test and validate medium-priority bug fixes (1 week)
- Task 2.5: Fix low-priority bugs (2 weeks)
- Task 2.6: Final testing and validation (1 week)

Ongoing:

- Task 2.7: Weekly review meetings (every week)
- Task 2.8: Continuous user feedback integration (ongoing)
- Task 2.9: Monthly progress review and reassessment (monthly)

Phase 2: Implementation of Multiple Streams (Months 4-6)

Month 3: Planning and Design

- Task 3.1: Define scope and specific features for subscriber and standard streams (1 week)
- Task 3.2: Designer develops a cohesive design plan (2 weeks)
- Task 3.3: Finalize design and feature specifications (1 week)

Months 4-6: Development of Subscriber and Standard Streams

Subscriber Stream:

- Task 4.1: Implement customizable training plans (3 weeks)
- Task 4.2: Develop heart rate-based training zones (3 weeks)
- Task 4.3: Enhance post-run analysis features (2 weeks)
- Task 4.4: User testing and feedback integration (2 weeks)

Standard Stream:

- Task 5.1: Improve goal flexibility (2 weeks)

- Task 5.2: Create educational content on metrics (3 weeks)
- Task 5.3: User testing and feedback integration (2 weeks)

Month 6: Marketing and Launch Preparation

- Task 6.1: Develop a marketing strategy (2 weeks)
- Task 6.2: Prepare promotional materials and content (1 week)
- Task 6.3: Roll out new features to users (2 weeks)
- Task 6.4: Communicate improvements and new features to users (1 week)

Ongoing:

- Task 6.5: Weekly development and design sync meetings (every week)
- Task 6.6: Continuous user feedback collection and integration (ongoing)

Activity	Description
A	Identify and categorise bugs
B	Prioritise bugs on impact
C	Develop bug resolution plan
D	Initial testing and verification of reports
E	Fix High priority bugs
F	Conduct thorough testing of fixed bugs
G	Fix medium priority bugs
H	Test and validate medium priority bugs
I	Fix low-priority bugs
J	Final testing and validation
K	Define scope and specific features for subscriber and standard streams
L	Designer develops a cohesive design plan
M	Finalise design and feature specifications
N	Implement customisable training plans
O	Develop heart rate-based training zones
P	Enhance post-run analysis
Q	User testing and feedback integration
R	Improve goal flexibility
S	Create educational content on metrics
T	Develop marketing strategy
U	Prepare promotional materials
V	Roll out new features
W	Communicate improvements to users

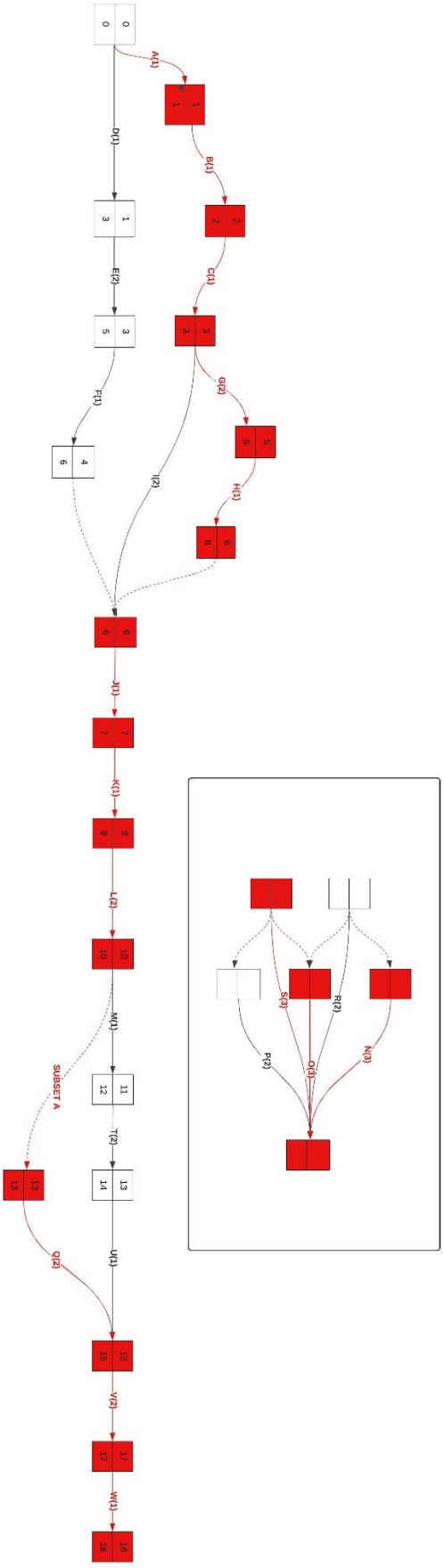
Activity	Dependencies
A	--
B	A

C	B
D	A
E	C
F	E
G	C
H	G
I	C
J	E,G,I
K	--
L	K
M	L
N	L
O	L
P	L
Q	P,O,N,M,R,S
R	L
S	L
T	M
U	T
V	Q,U
W	V

Activity Network:

0
0
A(1)
1
1
D(1)
B(1)
1
3
2
2
E(2)
C(1)
3
5
3
3
F(1)
G(2)
4
6
5
5
I(2)
H(1)
6
6
6
6
J(1)
7
7
K(1)
8
8
L(2)
10
10
M(1)
SUBSET A
11
12
T(2)
13
13
13
14
Q(2)
U(1)
15
15
V(2)
17
17
W(1)
18
18
S(3)
R(2)
P(2)
O(3)
N(3)

Critical Path analysis:

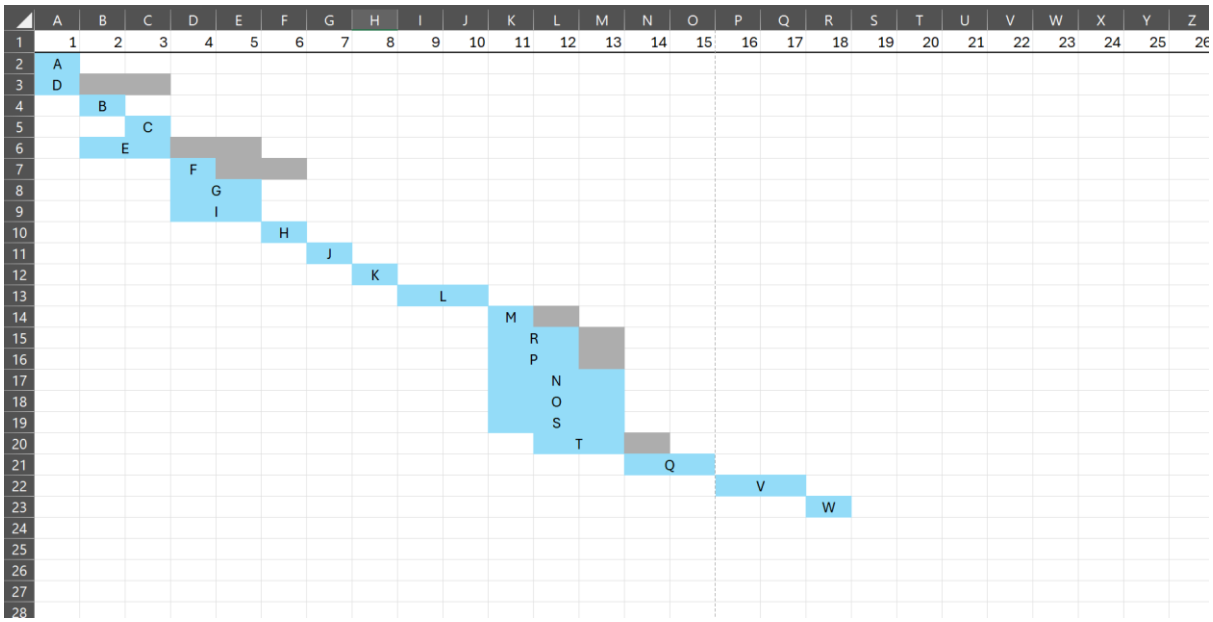


The critical paths for this network are:

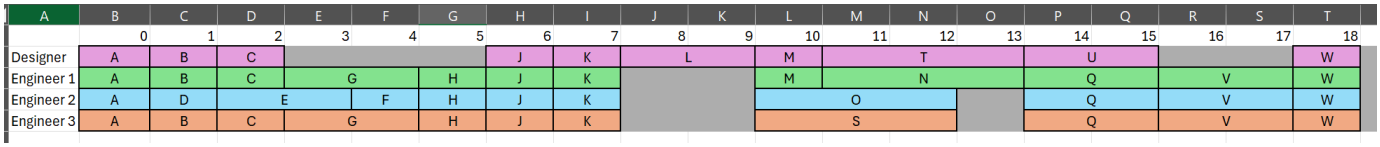
A,B,C,G,H,J,K,L,(S,O,N)[1]*,Q,V,W*

The weight is 18 weeks

Gantt chart:



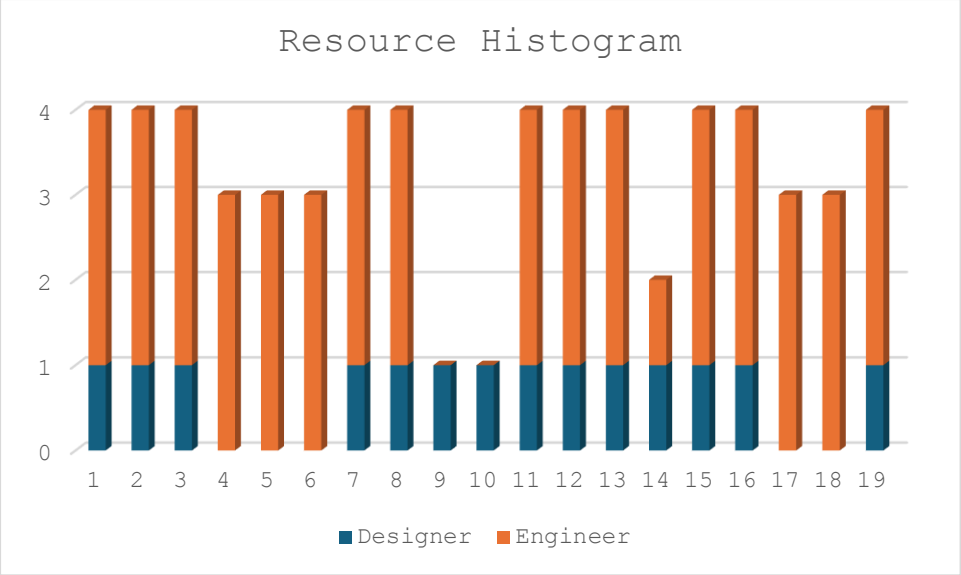
Scheduling diagram:



As can be seen from the scheduling diagram the project can be completed pretty optimally with only a single extra day being needed to accommodate all of the critical tasks with the designated staffing. However, there are two clear choke points. This could be improved by replacing one of the engineers with another designer. This scheduling diagram fails to consider the separate streams, meaning that one engineer might go from one stream to another without consistency. This shouldn't be too much of a problem. At the end of scheduling, there are 8 weeks float on the project, these could be placed throughout to provide some leeway with timings if needed as well as to alleviate the crush of the deadline as the current system is very busy

Resource histogram:

[1] Any of the following activities produces a valid critical path



Stakeholder Presentation:

RUNWIZE FOR THE FUTURE

Tom piercy

KEY CLIENT FEEDBACK

Feedback	Severity
Reminders on the right day	High
Better privacy and safety	High
Ability to set own goals	Medium
Better UI/Help menu to interpret data	Medium
More feedback post-run	Medium
Setting other targets	Medium
Personalised reminders	Medium
Missing advanced training features	Low
Heart-rate insights	Low
Community interaction	Low

User feedback is crucial for improvement and success of any product or service. For an app like RunWize, user feedback provides direct insights into the user experience, helping to identify pain points, areas for enhancement, and new feature opportunities.

Listening to users ensures that the app evolves to meet their needs and expectations, fostering loyalty and reducing churn. It also allows the company to stay competitive by quickly adapting to trends and user demands.

CHOSEN SOLUTION

1 Shared goal

2 Development phases

3 months, with technical audits to manage time appropriately

DEV PHASE 1

- Comprehensive bug auditing
- Bug resolution
- User Feedback integration
- Review and assessment

DEV PHASE 2

- Planning and design
- Stream development
- User testing and feedback
- Marketing and launch

PERFORMANCE METRICS

Phase 1	Phase 2
Bug Resolution Rate	Monthly Active Users (MAU)
App Stability	Paid Subscriber Growth
User Feedback on Bugs	Feature Usage Rate
	User Satisfaction
	Feature Adoption Rate

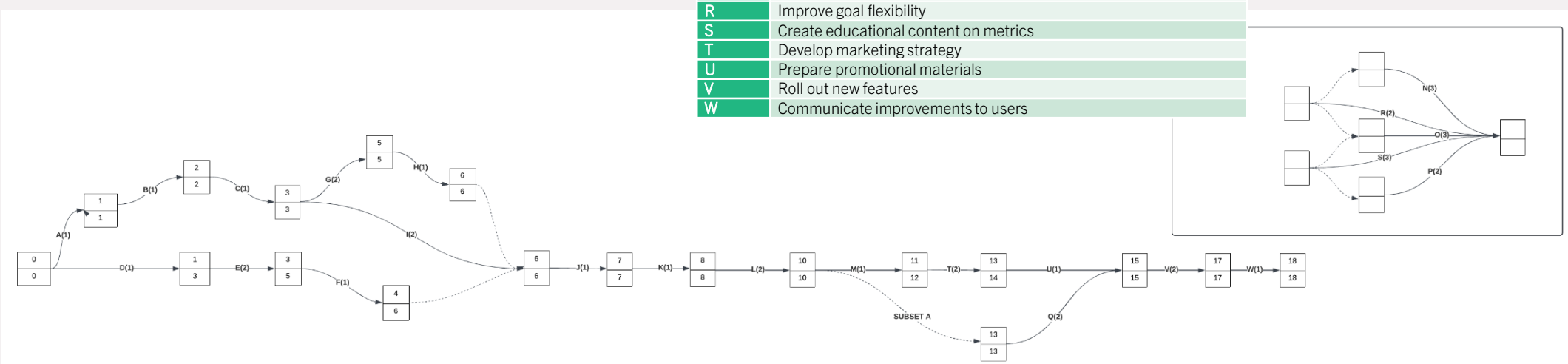
TECHNOLOGY METHODOLOGY

Agile

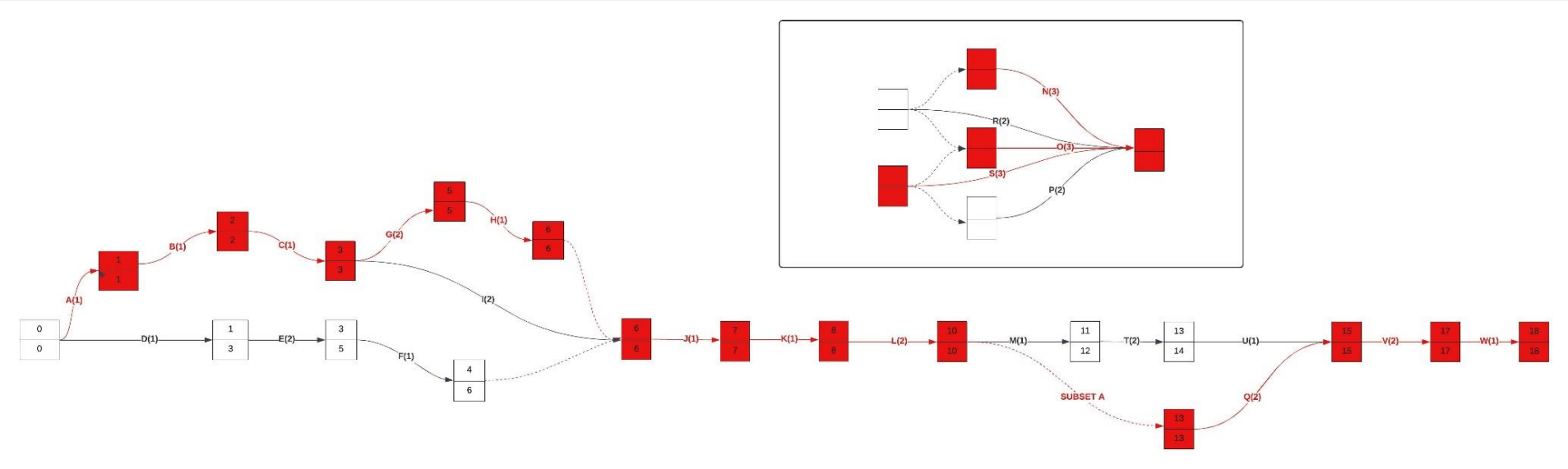
- Adaptability
- Incremental progress
- Collaborative approach
- Flexibility

ACTIVITY NETWORK

Activity	Description
A	Identify and categorise bugs
B	Prioritise bugs on impact
C	Develop bug resolution plan
D	Initial testing and verification of reports
E	Fix High priority bugs
F	Conduct thorough testing of fixed bugs
G	Fix medium priority bugs
H	Test and validate medium priority bugs
I	Fix low-priority bugs
J	Final testing and validation
K	Define scope and specific features for subscriber and standard streams
L	Designer develops a cohesive design plan
M	Finalise design and feature specifications
N	Implement customisable training plans
O	Develop heart rate-based training zones
P	Enhance post-run analysis
Q	User testing and feedback integration
R	Improve goal flexibility
S	Create educational content on metrics
T	Develop marketing strategy
U	Prepare promotional materials
V	Roll out new features
W	Communicate improvements to users



PROJECT ANALYSIS



PROJECT ANALYSIS CONTINUED

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Designer	A	B	C				J	K	L		M	T			U				W
Engineer 1	A	B	C	G		H	J	K			M	N			Q		V		W
Engineer 2	A	D	E		F	H	J	K			O				Q		V		W
Engineer 3	A	B	C	G		H	J	K			S				Q		V		W