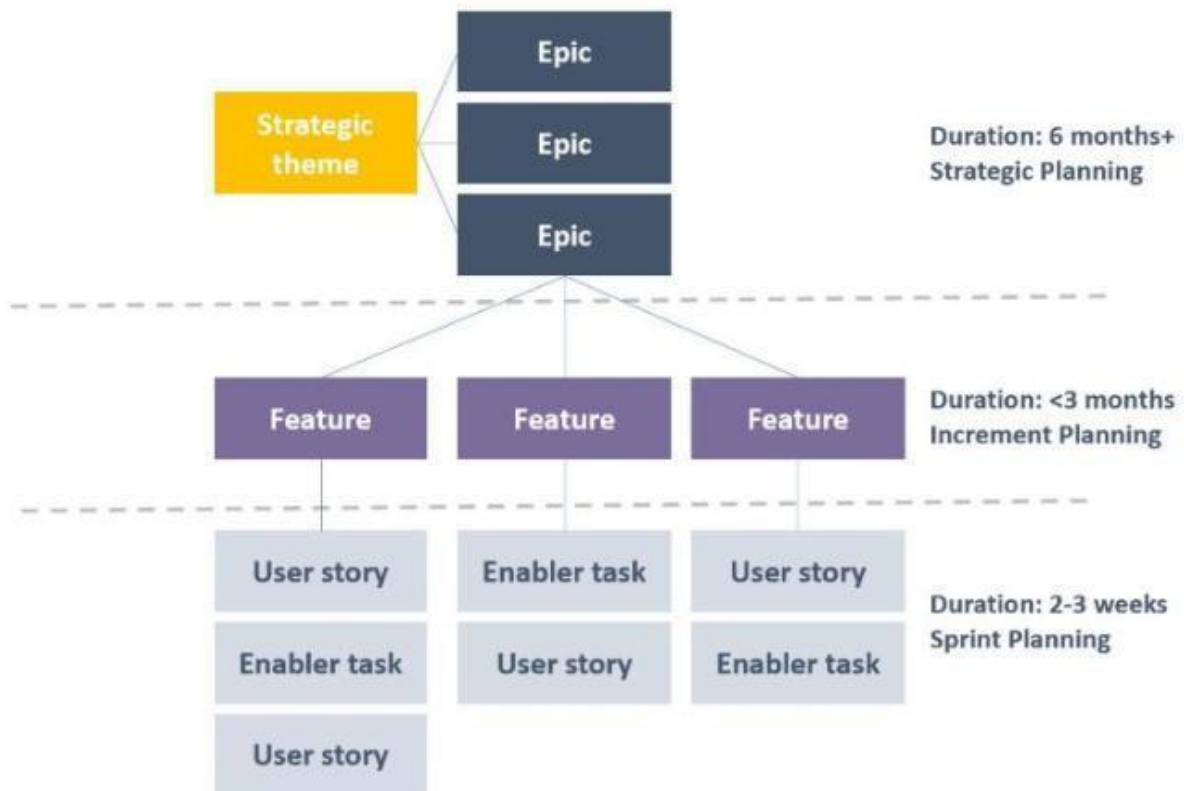




Epics Features and Stories

Fig: Breaking down the work



Epics

What is an Epic?

An Agile Epic is a large body of work that will be delivered over multiple sprints. Often supported by a business case, they are significant pieces of work that strategically add value. Epics help organisations break their work down, organise that work, while continuing to work towards a bigger goal.

In a sense, epics in agile are similar to epics in film or literature. Epics can be broken down into specific pieces of work, called Features. These are based on the needs and requests of customers or end users and is sized or split as necessary to be delivered by the Agile teams.

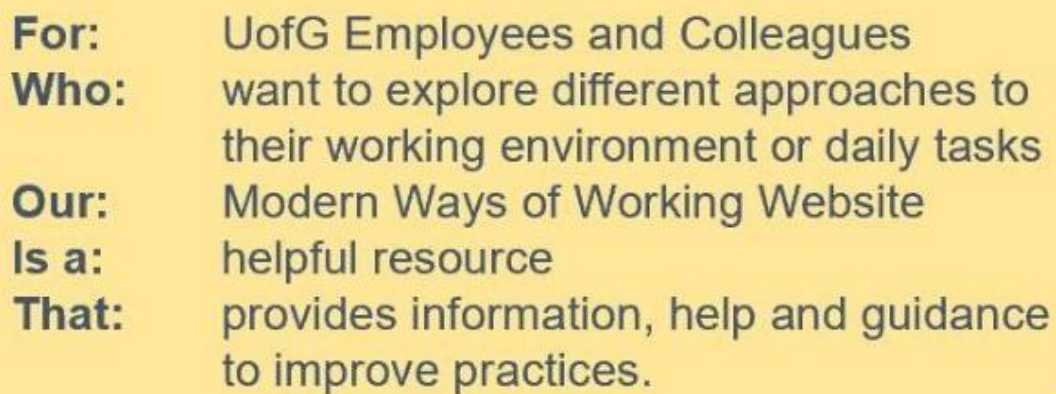
Epics are a helpful way to organise your work and to create a hierarchy. The idea is to break work down into deliverable pieces, so that large projects can actually get done and you can continue to deliver value on a regular basis.

For example, the design, creation, testing and delivery of a transactional website or app, or complex training programme is about the right size for an Epic.

Writing an Epic

A well-written epic is a key to have a good understanding and material to refer in case of any doubts during the development work. It helps in avoiding a lot of conflicts and misunderstanding in the team and with stakeholders. Since this is what you will refer to when breaking down the work it's extremely important to collaborate when developing your Epic.

As a simple guide, the main elements of an Epic include the user, the product and design requirement, expressed as a story that encapsulates the future state. A simple way of structuring an Epic is as follows:



For:	UofG Employees and Colleagues
Who:	want to explore different approaches to their working environment or daily tasks
Our:	Modern Ways of Working Website
Is a:	helpful resource
That:	provides information, help and guidance to improve practices.

Features

What is a Feature?

A feature is a chunk of work from the Epic – a deliverable that adds value and moves towards completing the Epic.

A feature should:

- provide business value
- it should be estimable – it must have enough definition for the team to provide an estimate of the work involved in implementing it
- be small enough to fit within 1 to 3 sprints – therefore, if it is too big, it should be broken down further
- be testable – you should understand what test a feature should pass in order to be acceptable to the customer.

Writing a Feature

As a minimum, the expression of a feature should contain a short descriptor of the item of value, a description of the benefit of the feature, and the acceptance criteria (the points of quality or completion the feature must achieve, UAC for short).

Feature: Agile Glossary

Benefit: Explanation of terminology to support working within an Agile environment.

User Acceptance

Criteria: Easy to understand, sharable, accessible.

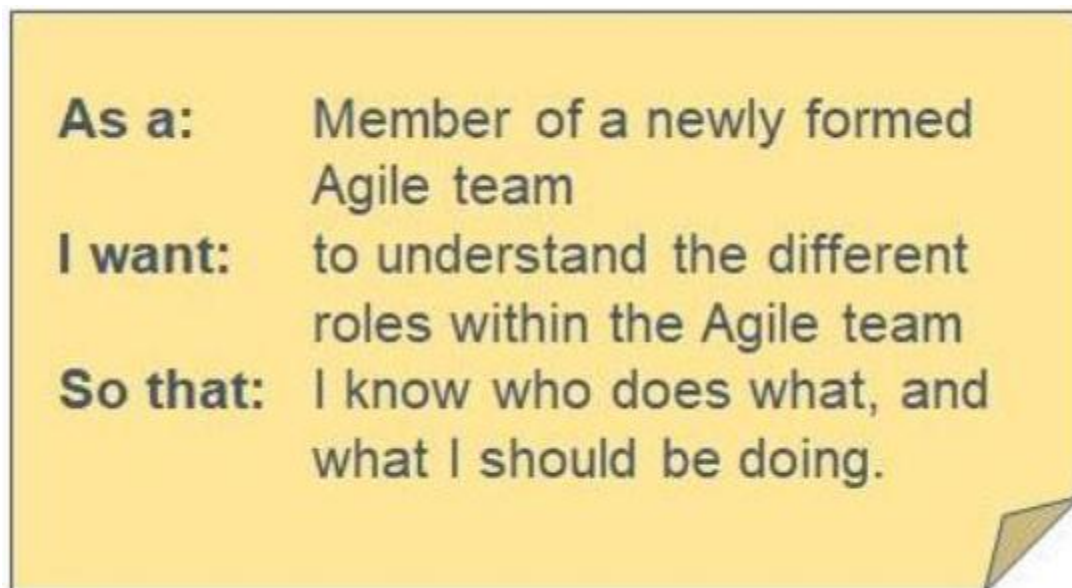
Stories

What is a Story

Stories, also called “user stories,” are short requirements or requests written from the perspective of an end user. They may also be enabler tasks that support the work of completing a feature. Stories are things that teams can usually complete within a single sprint.

Writing a Story

A Story should describe a need that can be satisfied by introducing a new feature or changing an existing feature. In other words, stories identify what someone wants to accomplish with your product and why.



Now that we know about Epics, Features and Stories let's look in more depth at Stories to see what else can be done to more effectively manage workload.

Story Splitting

What is story splitting?

Story splitting is the practice of breaking big pieces of work down into smaller, deliverable pieces of work.

Why do I need to split my stories?

Splitting stories helps teams to understand the tasks and complexities involved in delivering a piece of work.

You may want to split your stories based on:

- Steps in a workflow
- Business objectives
- Overcoming user problems
- Data
- Anticipated user behaviour
- The 'stuff we know' from the 'stuff that needs more research'
- Operations and interactions
- Effort
- Complexity

All of this is explained in much more depth, with examples, [on the Agile For All website](#).

How do I split my stories?

Think about all of the different tasks you need to do to complete a story and break these out into stories in their own right.

There are many patterns to help you do this, and many ways to ensure your stories retain their quality as you break them down. Use the [INVEST principles](#) to guide you.

For example:

Big story

- As a Head of Admin, I want to publish a piece of content to the UofG website about my department.

Small, split stories

- As a Head of Admin, I want to draft a piece of content about my department
- As a Head of Admin, I want my content to be engaging and visual
- As a Head of Admin, I want my peers to review my content

Story Mapping

What is story mapping?

Story mapping is a method for arranging user stories to create a more holistic view of how they fit into the overall user experience.

What are the benefits of story mapping?

Story maps:

- help develop shared understanding within and across teams
- make estimating more realistic because everyone has a better understanding of all of the tasks it takes to complete a piece of work
- help visualise dependencies
- help visualise and capture ideas as well as tasks
- make it easier to prioritise tasks and outcomes

Visualising stories

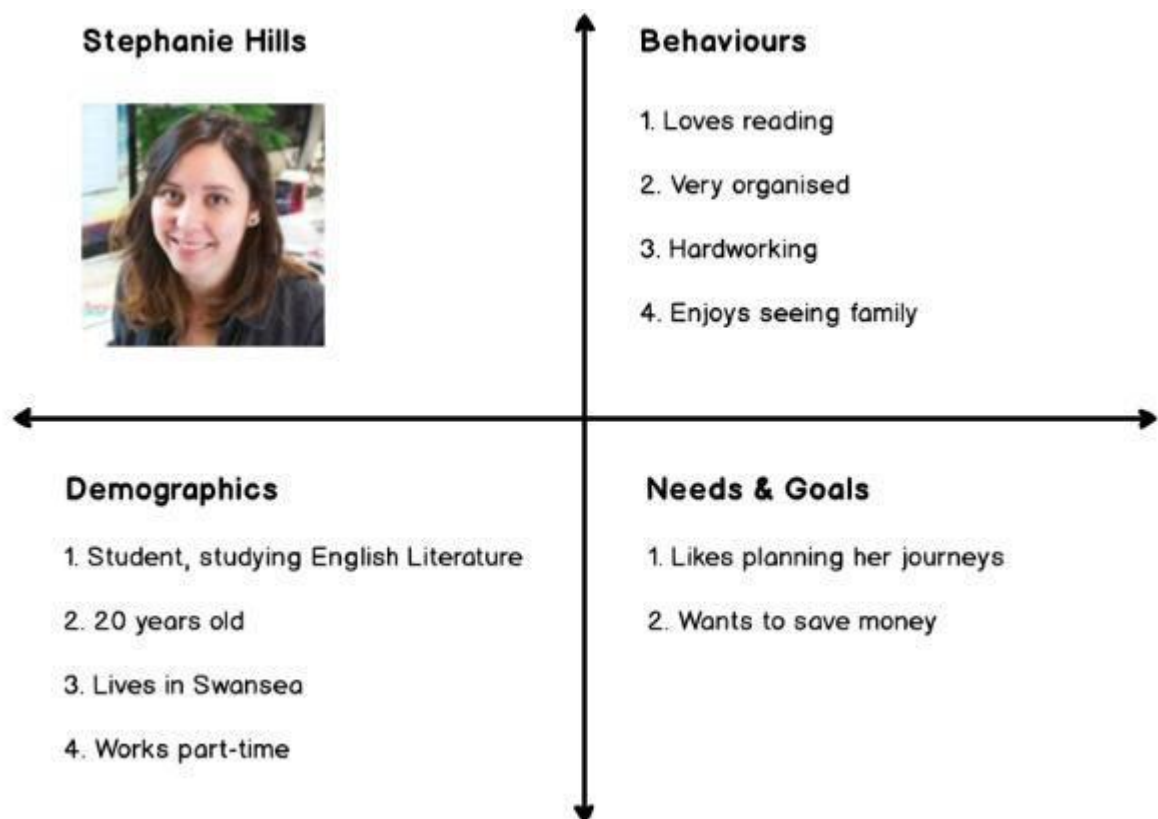
When working with your team to build story maps, try creating simple, visual prompts to help support your discussions. You could do this on post-it notes or small cards. The trick with this is to write down everything!

One thing that goes wrong a lot when you are having discussions about work to be completed, is ideas get lost, so write these down alongside your work. You can always get rid of them later if they are no longer relevant.

Step by step guide

Users

Start with your personas or user insights. Place these at the top of your map.



Activities are the common goals and the steps that belong to the user journey. You will find these in the 'needs' part of your personas.



Can compare
ticket prices
across different
transport
methods

Tasks/User Needs

Tasks or User Needs provide a bit more narrative on the tasks on the activities above. These should flow left to right and indicate user journeys or behaviour.



Activity

Can compare
ticket prices
across different
transport
methods

User need

Can filter
by bus,
train and
taxi

User need

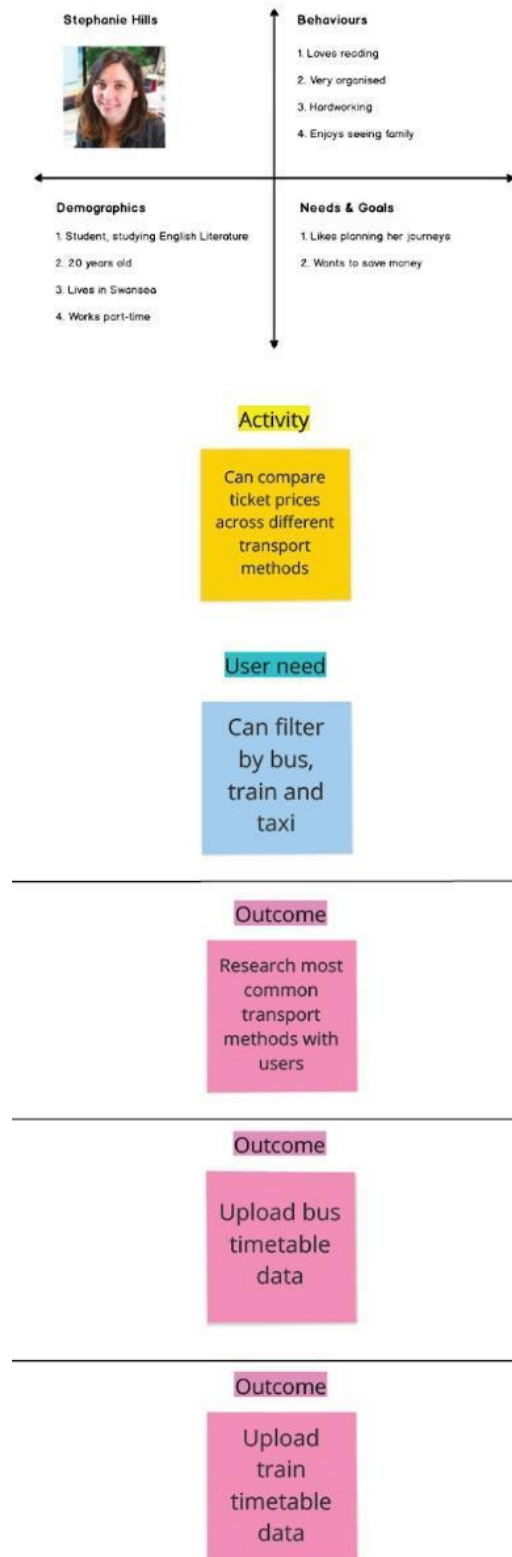
Can see all
ticket options
side by side
with journey
times

User need

Can select
ticket and
book online

Outcomes

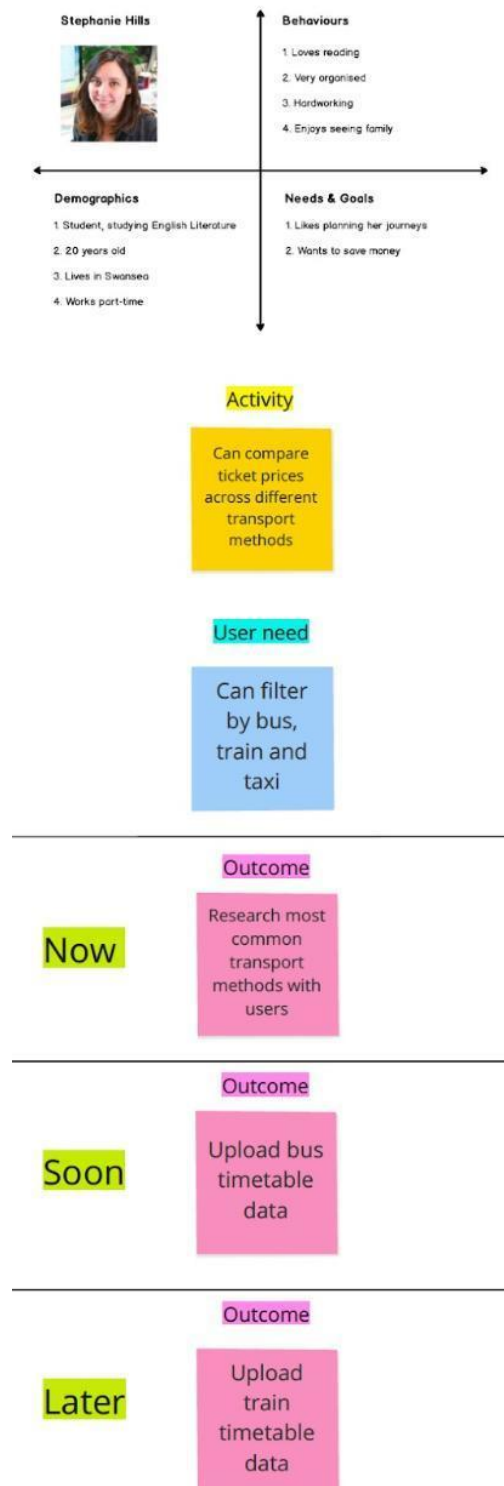
Outcomes are the stories that start to explain chunks of work that need to be completed. These can be articulated as features, epics, or jobs to be done, whatever works best for the team. Map these down the way, from the task/user need they are associated with.



Timescales

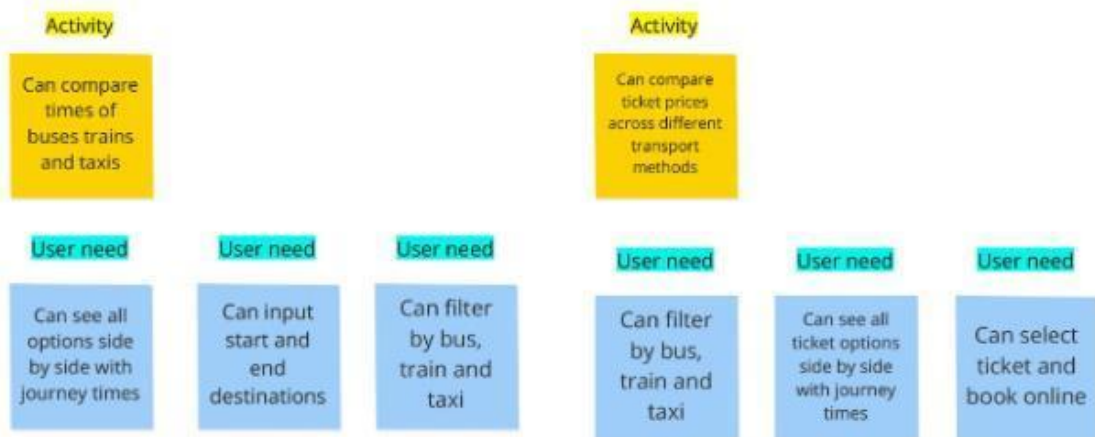
Don't get too hung up on timescales, but use your map to begin a conversation about 'when'.

This should get the team talking about priorities. If it is too soon to make a decision, use the 'now, soon, later' format.



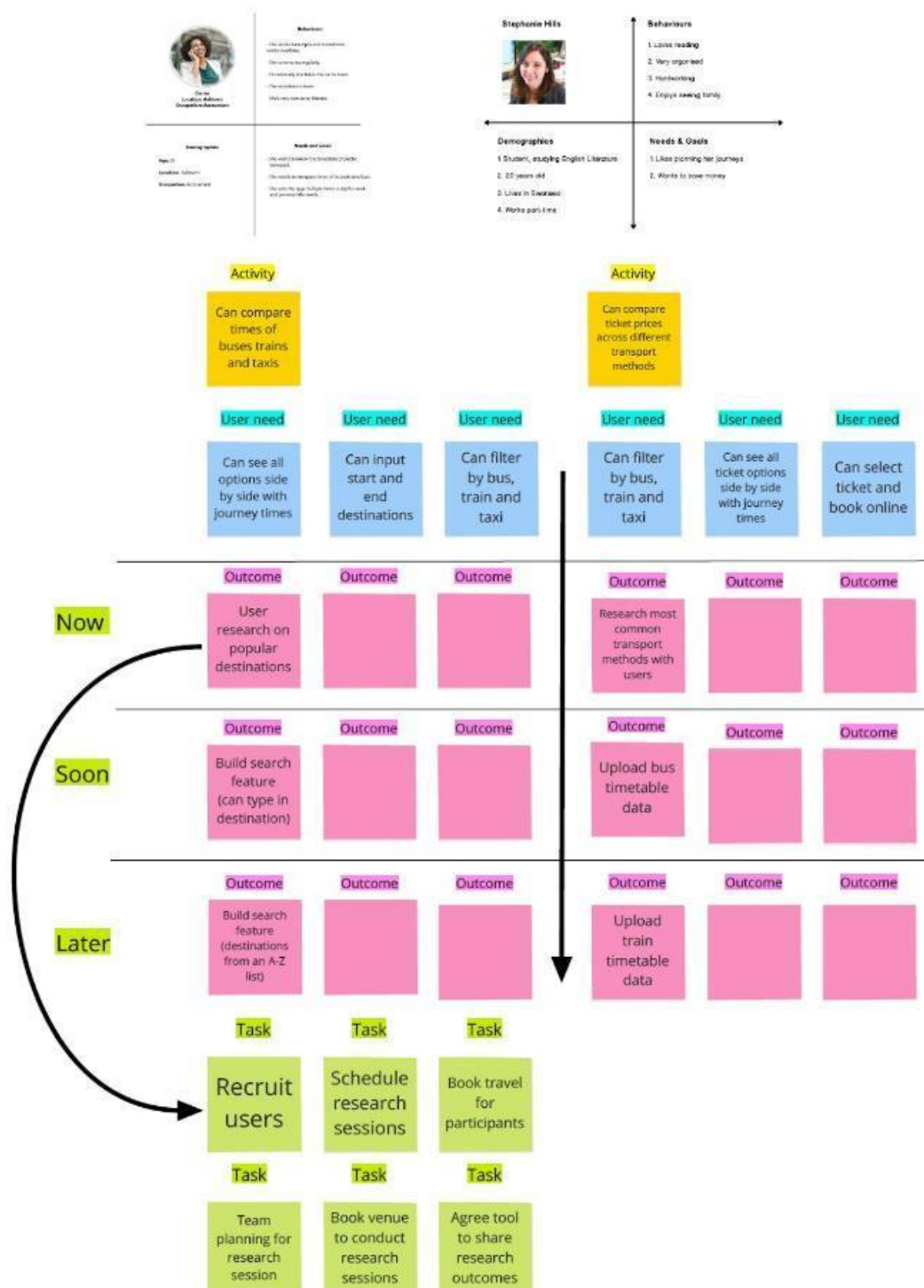
Build on it

Add your other personas to get a full view of all your priorities.

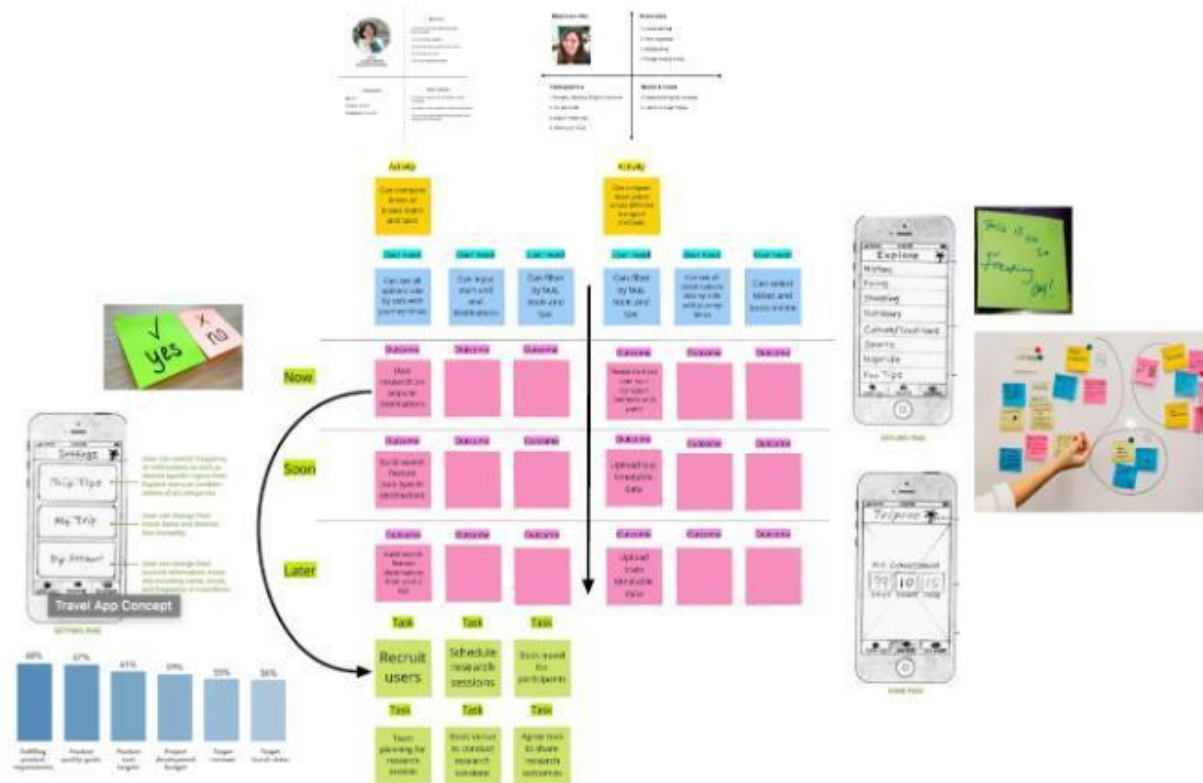


Tasks

You should now be able to get a clearer view of what stories and tasks you might want to take into your team's backlog.



Around your map should be product or service goals to keep people focused, sketches, storyboards, wireframes, sticky notes with feedback from users and stakeholders - anything of relevance.



There is no right or wrong way to story map

Remember ultimately that stories are just a means of shared understanding, and there is no concrete way to write them correctly or incorrectly. The most valuable part of story mapping as a team is to develop shared understanding

How can we help?

If you need some advice on get started with story mapping, the Responsive Solutions team can help.

[Get in touch](#) with us to explore the ways we can help.