AGILE PROJECT

FACULTY OF ENGINEERING
AIN SHAMS UNIVERSITY
International Credit Hours Engineering Programs (i.CHEP)

Project Document

Due Date:

01/12/2022

Submitted to:

Dr. Salwa Osama

Eng. Rana Sobhy

Submitted by:

Omar Ashraf Mabrouk

Zakaria Sobhy Abd El-Salam

Mahmoud Mohamed Seddik

Mahmoud Mohamed Omar

Abdelrahman Mohamed Salah

Course Information

ASU Course Code	***ASU Course Name***
CSE 233	Agile Software Engineering
Semester	**Due Date**
Fall 2022	1-1-2023

Participants Information

Full Name	**ASU ID**
Omar Ashraf Mabrouk	19P8102
Mahmoud Mohamed Omar Ibrahim	19P5803
Zakaria Sobhy Abd El-Salam Soliman Madkour	19P2676
Mahmoud Mohamed Seddik	19P3374
Abdelrahman Mohamed Salah	19P9131

Contents

Agile Methodology

Agile is an iterative approach to project management and software development that helps teams deliver value to their customers faster and with fewer headaches. Instead of betting everything on a "big bang" launch, an agile team delivers work in small, but consumable, increments. Requirements, plans, and results are evaluated continuously so teams have a natural mechanism for responding to change quickly.

It is based on 4 key principles called the agile manifesto:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Client Report

Customer Requirements

- E-commerce capabilities – payment gateways for donations
- Social media integration, allow easy access to the charity's pages on different social media platforms.
- Contact forms to volunteer or make donations in person.
- Describe what the charity organization does, this should be included anywhere on the homepage, as such information must be easy to find.
- It must be highlighted in the homepage that users can help in many ways as well (other than donating), either by volunteering or by promoting the charity site on social networks to help spread the word about your cause, and more.
- The website must focus on showing how their contribution will be used and how they're making a difference to someone's life.

Customer Questions

The following are customer questions regarding the developed websites and their answers:

1. **How much will the development process cost?**
   This project is completely free of charge as a contribution towards the goals of this organization.
2. **How long will the development process take till delivery?**
   It will take approximately 2 months to finish the websites.
3. **When will be the first time I can view the website?**
   An initial prototype is developed to involve the customer in the design process.

4. **Can a change be made to the website if required?**
   Yes, obviously one of the many benefits of agile software development is that we can change requirements, design and development as needed.
   "Responding to change over following a plan"
5. **Will the project have sufficient documentation for future use?**
   Working software over comprehensive documentation" means that delivering software that does what it should come first in the priorities before documentation. However, it doesn't mean that you should not create documentation; it means you should create documentation that provides value and at the same time does not hinder the team's progress.

Actions

In order to deliver our website to the customer we had two releases each having its own set of features which can be published regardless of work that hasn't been done yet.

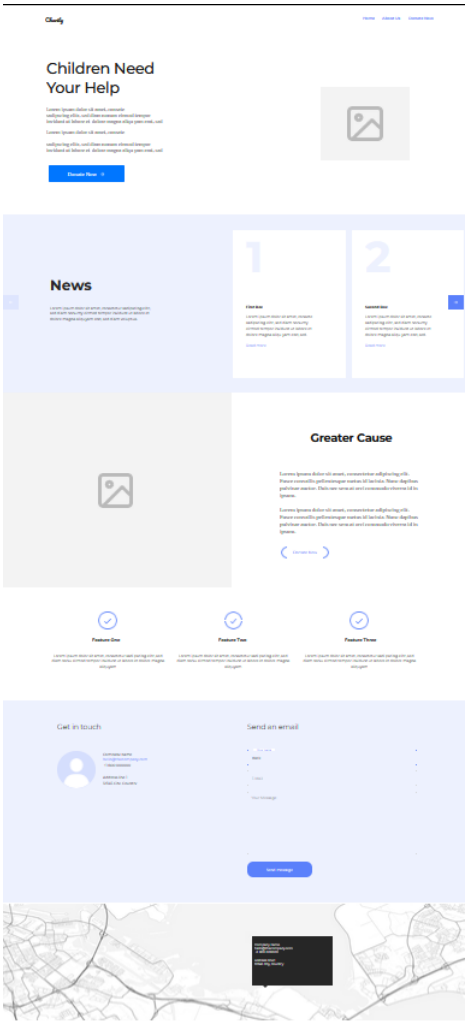
Release 1

1. First we had to start with the registration process:
   - Designed User registration page
   - Designed User login page
2. Then we started developing the admin dashboard and related pages:
   - Designed Admin page
   - Allow admin to view user details
3. Developing home page:
   - Adding button to redirect users to the organization's social media pages.
   - Form to send message to the organization.
   - Sufficient information about the organization's campaigns to be provided.
4. Donation payment page:
   - Design the donation payment page

Release 2

1. Regarding user registration:
   - User sign up information storage.
   - User Login authentication.
   - All user registration backend completed.
2. Admin Dashboard:
   - Admin able to change user details (changes stored in database).
   - Fetch donation details from database for admin to view.
3. Donation payment page:
   - Allow different payment options.
   - Authenticating user payment card details

Prototype



Since Home Screen is one of the most important aspects of a website, the prototype was based on it.

The home page shall be divided into 6 main sections: quick business into with charity mission and vision and a donate button, the second shall be a slide show with different campaign titles and descriptions.

the third shall be focused more on company mission and trigger compassion for the reader to donate, the forth part shall be the contact form, where user could contact site admin about any issue/concern.

The fifth part shall be about charity location from google maps and finally, the footer section will have different links to social media profiles, and copyright statement.

Creating a prototype makes it possible to concretize an idea and assess which features pose difficulty in implementation. Prototyping can thus identify unanticipated physical, technical, or financial constraints.

Enhance Website Quality

A well-designed prototype will enable you to:

- Conduct testing for site usability
- Inspect site navigation
- Conveniently access information on the site
- Determine correct placement of visual accents – what visitors should see first

Effectively Present Idea to Customers

Prototyping makes it possible to present your future product to potential customers before the actual launch of the product. It could also allow you to devise your marketing strategies better and start pre-sales.

Reduced Risks

Projects with a complete prototyping process are at lower risk than projects without prototyping. This is because prototyping directly affects project resources, time, and budget. Through prototyping, it is possible to estimate the resources needed and time for development.

Iterate at Lower Costs

Information gathered from potential customers through prototyping makes it possible to improve the product until an optimal product is formulated. A good idea can be to create several prototypes before the launch of mass production so that the additional costs of unsold products and reprogramming can be curtailed.

Simulate the Future Product

The most important advantage of prototyping is that it creates a model of the final product. It can help lure customers to invest in the product prior to any resource allocation for implementation. You can discover design errors and check their correctness before going into production.

Provide Focused Feedback

Exposing the prototype helps to get focused customer feedback on the desired qualities in the product. This feedback is critical to understand the needs and expectations of users, business requirements and gain a clear idea of what the product is headed for.

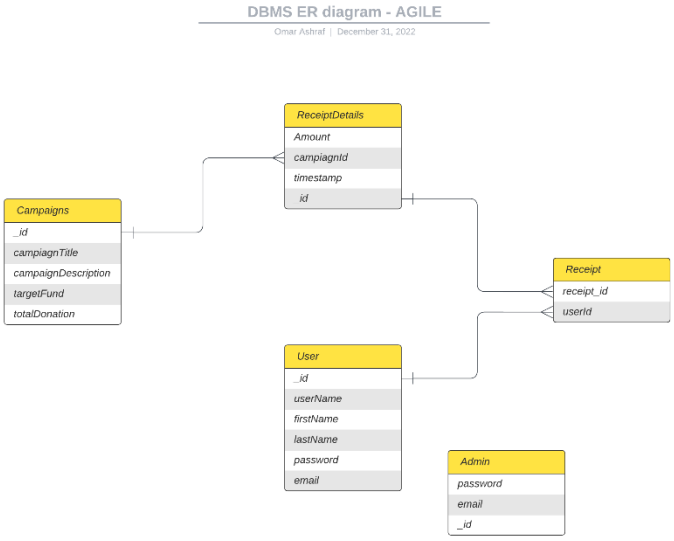
Planning

Through prototyping, the design team gets essential information that helps them to plan out the implementation. A prototype helps build user stories and emphasize on user needs. This brings substantial benefits to the scrum teams.

Quick and Easy

A designer can quickly develop a ready-to-implement prototype even from a simple idea on paper if they understand the logic and functionality of the product.

DB Schema



The database schema is based upon the 4 core key classes of agile which are product, user, receipt details and receipt. Each of them corresponds to campaigns, user, ReceiptDetails and Receipt as shown in the figure above.

A good database design has many benefits:

Easy Retrieval of Information

If the design is developed properly, then it would be easier to retrieve information. Correct design means the tables, constraints, and relationships created are flawless.

Easier Modification

Changes that you make to the value of a given field will not adversely affect the values of other fields within the table.

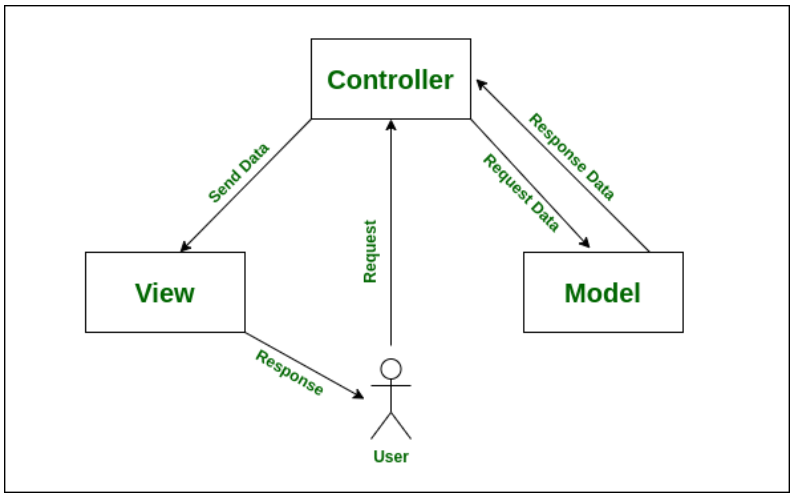
Easy to Maintain

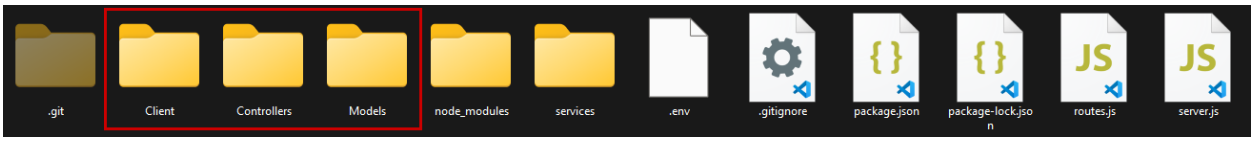
The database structure should be easy to maintain. The design is perfect if changes in one field is not affecting changes in another field.

Information

With a good design, you can enhance the quality and consistency of information.

MVC Architecture





The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.

MVC Components

Model

The Model component corresponds to all the data-related logic that the user works with. This can represent either the data that is being transferred between the View and Controller components or any other business logic-related data. For example, a Customer object will retrieve the customer information from the database, manipulate it and update it data back to the database or use it to render data.

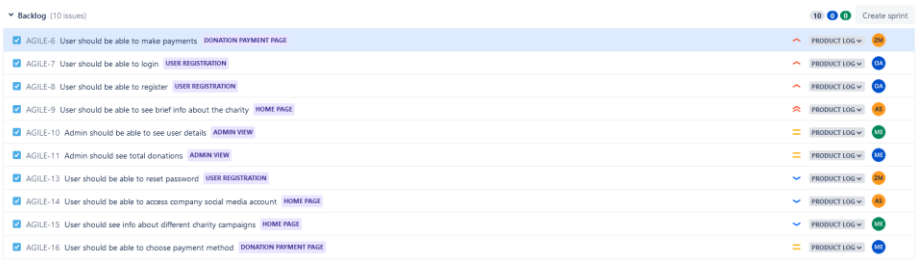
View

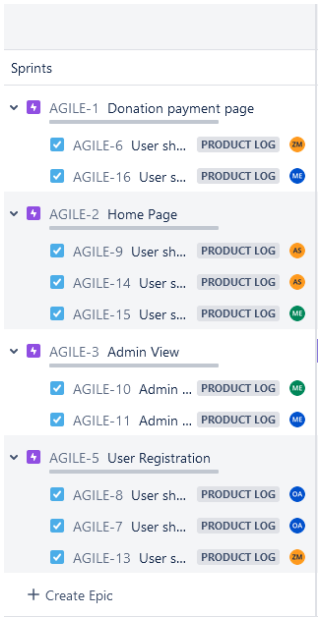
The View component is used for all the UI logic of the application. For example, the Customer view will include all the UI components such as text boxes, dropdowns, etc. that the final user interacts with.

Controller

Controllers act as an interface between Model and View components to process all the business logic and incoming requests, manipulate data using the Model component and interact with the Views to render the final output. For example, the Customer controller will handle all the interactions and inputs from the Customer View and update the database using the Customer Model. The same controller will be used to view the Customer data.

JIRA Account and Our Backlog





What is a Backlog?

A backlog is a list of tasks required to support a larger strategic plan. For example, a product development context contains a prioritized list of items. The product team agrees to work on these projects next. Typical items on a product backlog include user stories, changes to existing functionality, and bug fixes.

One key component that gives a backlog meaning is the prioritized items. Therefore, the items ranked highest on the list represent the team's most important or urgent items to complete.

Why is it Important When Using Agile?

Agile primary strengths lie in rapidly delivering value to customers. Quick iterations and deployment of new functionality and enhancements keep the focus squarely on delighting customers. Moreover, this iteration helps achieve product priorities.

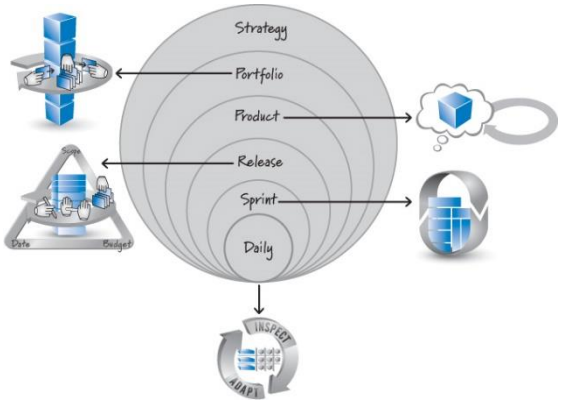
Sprint planning sessions rely on the backlog to scope, size, and slot development tasks and references. Furthermore, the development team will struggle to assess possible and create a reasonably confident schedule without these details captured in a single repository.

Consequently, product development teams may complete sprint tasks more quickly than expected. For example, particular projects may get unexpectedly put on hold or canceled. An accurate backlog allows the team to move swiftly. They can then focus on the following most essential items in the queue. The backlog prevents teams from idling. It gives them extra time on projects that aren't a priority.

A view into the backlog can also provide a preview of what's to come. It allows technical teams to begin thinking about how they might implement those items. Moreover, they can mitigate any conflicts, dependencies, or advanced work required. With a well-maintained backlog, the contents of any sprint will rarely be the first time the team has encountered the item and its requirements.

Multi-Level Planning

Agile does lot more planning and risk mitigation than traditional processes. Agile focuses on planning very often instead of doing comprehensive and assumption based planning once. Agile Planning (a.k.a. planning multi-level planning) has 6 levels - Strategy, Portfolio, Release, Iteration, Daily, and Continuous. Each layer of the multi-level planning drives the goals of the layers below, sets timeframes, and defines ownership and interaction.



Strategy – Strategic vision is at the top of the planning multi-level planning because it defines what a company is, and what it wants to become. The Timeframes on this layer are more about the individual strategic goals that are produced here. Typically, companies provide 1 year, and 5 year plans and share strategic vision and objectives with key management to pass down the chain. The layer is owned by C-Level management who define and govern all the execution of the strategic goals.

Portfolio – The portfolio layer of the multi-level planning represents the overall product offering that consists of application suites and tools and how they interact with each other. While our clients utilize individual products, they typically engage many products that integrate and work together to provide an overall solution. The owner of this layer is typically upper management in the product area and has vision into the various product lines. Decisions here should support the strategic vision and goals, and set direction for the product team's road-maps and integration synchronization.

Product – In our case, the product layer represents product teams who own one or more related products (product suite). Each team sets a product vision and outlines the road-map for their products within the product suite. The typical time-box for our products is approximately 12 months out. Product Managers own this layer of the planning multi-level planning and validate road-maps for products to the Portfolio and Strategic vision. A product roadmap is a high-level plan that describes how the product is likely to grow. It allows you to express where you want to take your product, and why it's worthwhile investing in it . An agile product roadmap also facilitates learning and change. A great way to achieve these objectives is to employ a goal-oriented roadmap – a roadmap based on goals rather than dominated by many features.

Release – A release represents a prioritized backlog of product features that represent smaller plans that drive toward the product vision. A release is time-boxed here into approximately year quarters such that there are four releases per year. This provides a means for prioritizing themes to target the product road-map. Product Managers typically own this layer and work with the Product Managers to prioritize and slot high-level themes into releases to create a plan for iterations.

Iteration – An iteration is a time-boxed set of features (stories) that we plan to deliver in two week increments. There are typically six iterations to a release. When planning, the product manager reviews the release plan and divides it into 4-6 iterations worth of backlog and prioritizes to deliver key features quickly. Using guidance from the release plan, Product and Product Managers determine approximate priority of themes and epics to meet release goals. Stories are initially created by the Product Manager and shared with the team in story workshops. The team commits to a set of stories each iteration in planning meetings.

User Stories

A user story is an informal, general explanation of a software feature written from the perspective of the end user. Its purpose is to articulate how a software feature will provide value to the customer

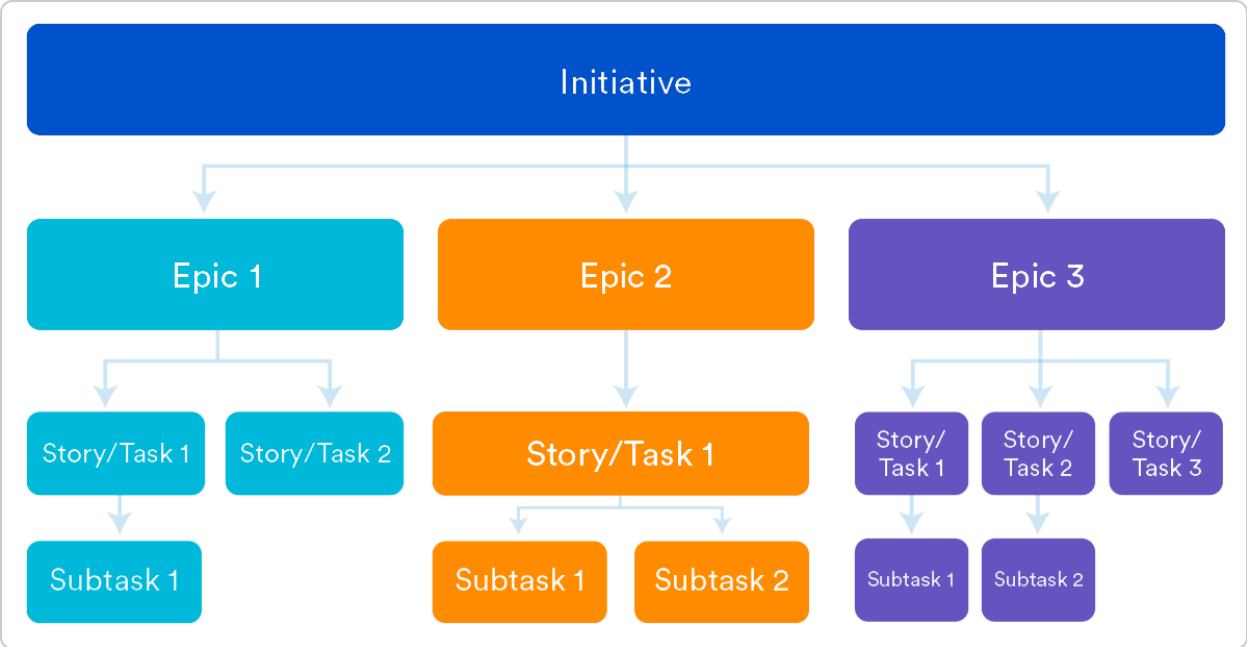
What are agile user stories?

A user story is the smallest unit of work in an agile framework. It's an end goal, not a feature, expressed from the software user's perspective.

A user story is an informal, general explanation of a software feature written from the perspective of the end user or customer.

The purpose of a user story is to articulate how a piece of work will deliver a particular value back to the customer. Note that "customers" don't have to be external end users in the traditional sense, they can also be internal customers or colleagues within your organization who depend on your team.

User stories are a few sentences in simple language that outline the desired outcome. They don't go into detail. Requirements are added later, once agreed upon by the team.



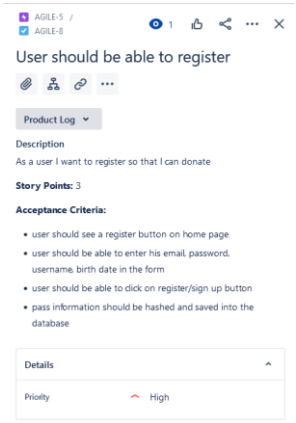
Why create user stories?

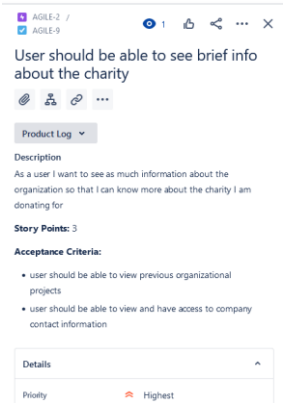
For development teams new to agile, user stories sometimes seem like an added step. Why not just break the big project (the epic) into a series of steps and get on with it? But stories give the team important context and associate tasks with the value those tasks bring.

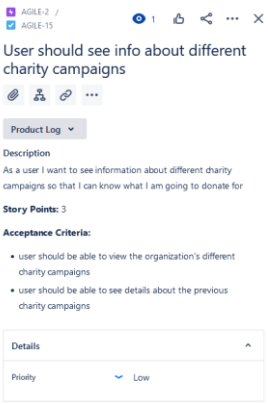
User stories serve a number of key benefits:

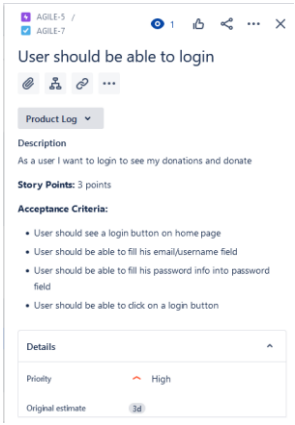
- Stories keep the focus on the user
- Stories enable collaboration
- Stories drive creative solutions
- Stories create momentum

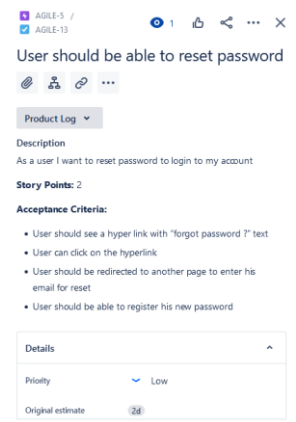
Our user stories:

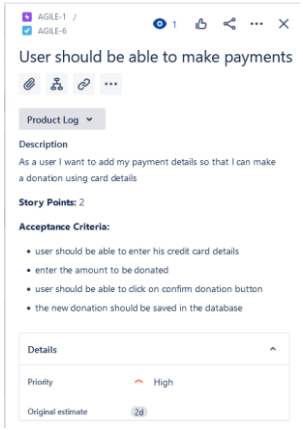


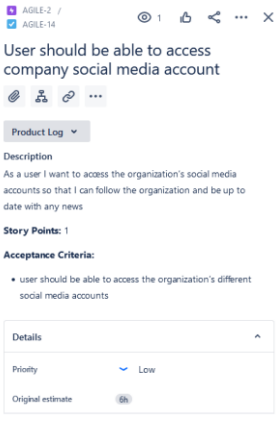


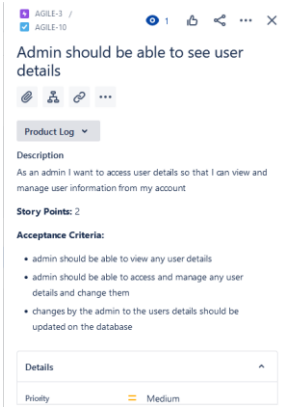


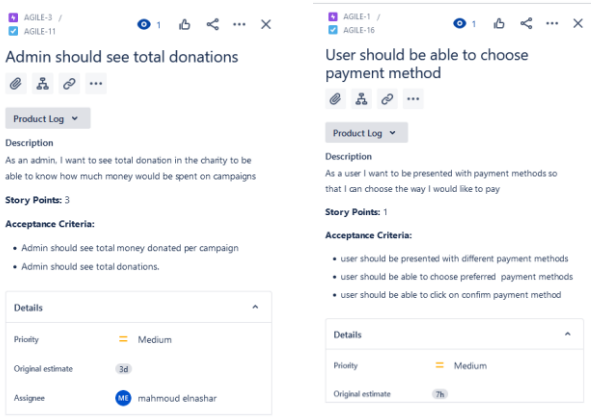












Story Map

What Is (User) Story Mapping?

Story Mapping or User Story Mapping is a technique used in product discovery: outlining a new product or a new feature for an existing product.

The result is a Story Map: all the user stories arranged in functional groups. This helps you keep your eye on the big picture while also providing all the details of the whole application.

What's the Purpose of Story Mapping?

The main purpose of Story Mapping is to facilitate product discovery and prioritization of development work. You achieve this by putting user activities and tasks on a map that serves to keep them in context.

The Story Map always shows how each individual story fits in the whole application. And this makes it easy to spot gaps and decide how important one is over another.

Story Mapping gives you a number of direct and indirect benefits.

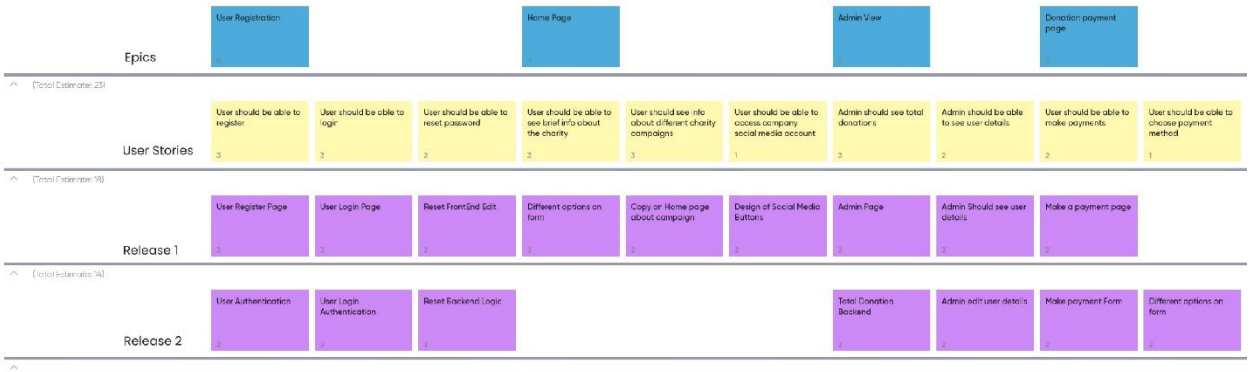
Everyone can easily understand the whole application—generally the most difficult part of software development. The Story Map tells the story of what your application solves and how it does it for anyone who's interested. Everyone can participate in creating it.

- You keep the big picture of your application in full view—losing the big picture is a common complaint in agile teams.
- Putting together and having a Story Map visible encourages iterative and incremental development.
- Shows you where a user story fits in the whole system in a single glance.
- Helps you decide what to build first. A Story Map makes it easy to pick and choose user stories from different features that together will provide meaningful

value. This means you can confidently determine the scope of and build an MVP or a useful release.

- You can more easily avoid building something that doesn't work. You won't get lost or forget important parts that would effectively render it unusable, like a car without brakes. For example, because you delayed low-value stories that your high-value stories depend on.
- Let you walk the Story Map to test it for gaps: spot more easily where something is missing.
- Let you make prioritization decisions taking the context of the whole system into account.
- You'll avoid staring yourself blind on a single user story.

Our Story Map:

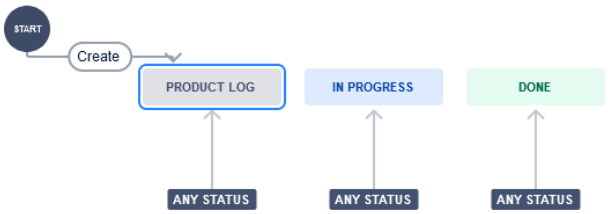


Agile Workflow

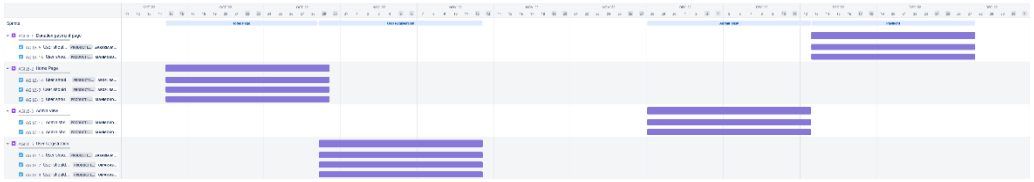
An agile workflow is a series of stages agile teams use to develop an application, from ideation to completion.

Every software team has a process they use to complete work. Normalizing that process–i.e., establishing it as a workflow–makes it clearly structured and repeatable, which, in turn, makes it scalable. At Atlassian, we take an iterative approach to workflow management because it helps us meet our goals faster and exemplifies our team culture. We are experts in agile workflow management (if we do say so ourselves), and we want to help you become experts too.

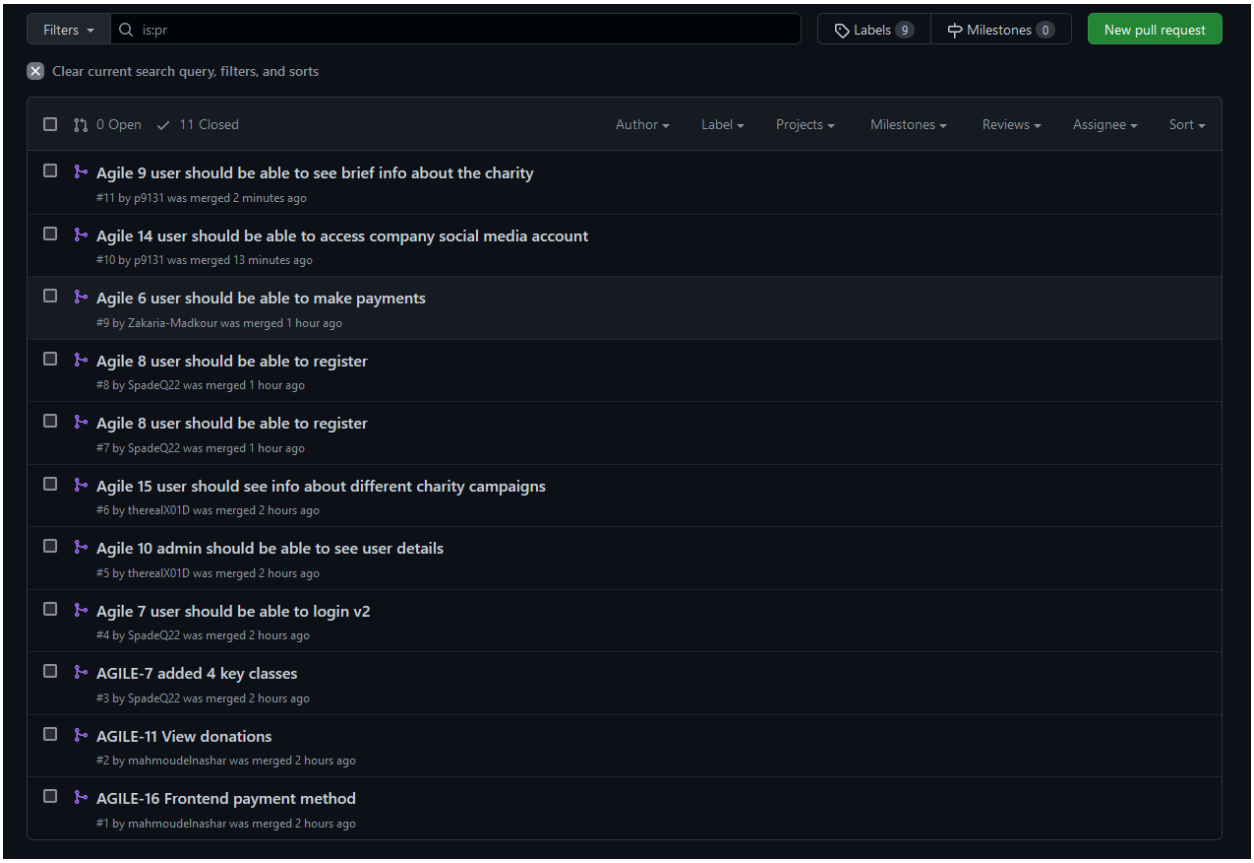
Our workflow:



Sprints:



GitHub



Clean Code

Example:

```
async function signUp(req, res) {
    //sign up function
    try {
        console.log(req.body);
        const newAdmin = await Admin.create(req.body);
        console.log(newAdmin);
        res.status(201).json(newAdmin);
    } catch (e) {
        res.status(500).json(e);
    }
}
```