System Implementation

(Running Scenarios)

Introduction

A research prototype system was developed to demonstrate the feasibility of the proposed UCD-Agile integration approach, which enhances the integration of UCD in distributed agile environments. Here we show validation of the UCD-Agile integration approach for three scenarios to ensure that a computer-based system is capable of demonstrating those scenarios.

The prototype implementation demonstrates that a computer system is able to perform the steps involved in each of the three UCD-Agile integration approach scenarios described in this document. The implementation for each scenario consists of a sequence of screens, where each screen represents one step of the scenario. Each screen displays the output that results from moving from one step to another.

The VisualStudio [11] was used to develop the prototype, and MySQL [12] was used as the backend database. Both are popular tools used for creating computer and web-based applications.

Scenario 1: Modifying the UI artifact

Scenario 1 Description

1. Developer1 modifies UI artifact codeId135.

2. UCD-Agile integration system identifies US5, US6, and US7 as potentially affected stories along with the status of each user story.

3. If at least one of the related user stories is ‘in-active’, then developer1 requests an approval from the affected team members (user story owner: ‘developer2’, UX-designers: ‘UX-designer2’) to proceed with the modification of the related UI code.

4. When the request is accepted by the affected member, developer1 will be notified and the user story status will be changed to ‘active’.

5. Developer1 modifies the UI artifact codeId135.

6. All other affected members will be notified automatically once the modification has been applied.

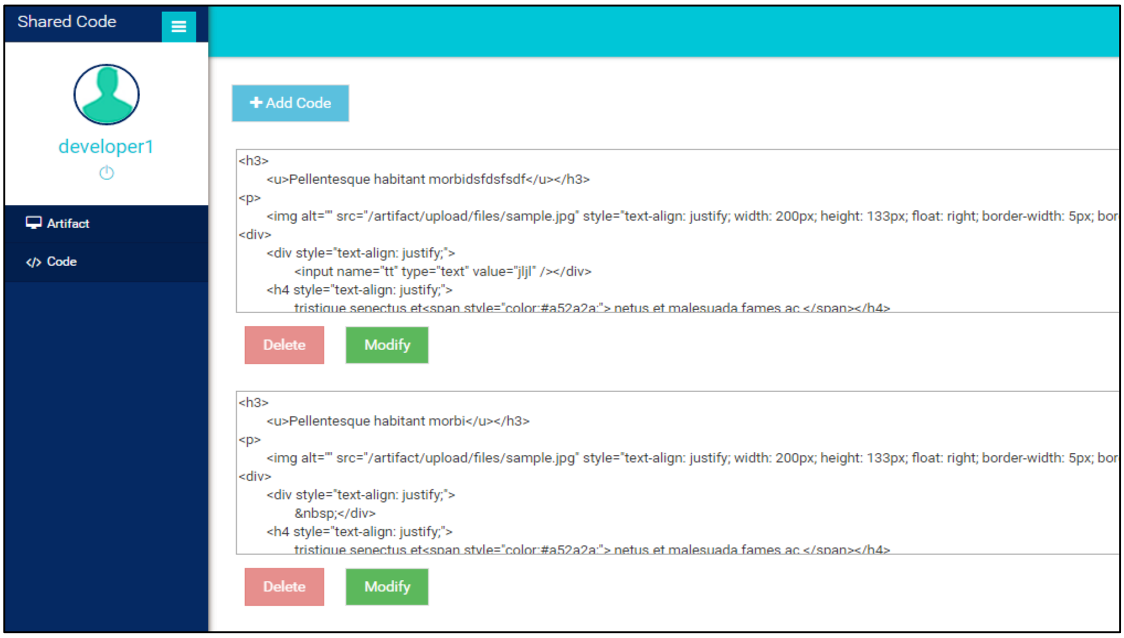
Running Scenario 1

Scenario 1 starts with the following initial data set:

- The UI artifact code codeId135 is shared to the following user stories: US5, US6, and US7.
- Some of the related user stories are ‘in-active’.
- Developer1 modifies UI codeId135.

The implementation of the six steps is discussed below.

1. Developer1 identifies codeId135 to be modified (Figure 1)

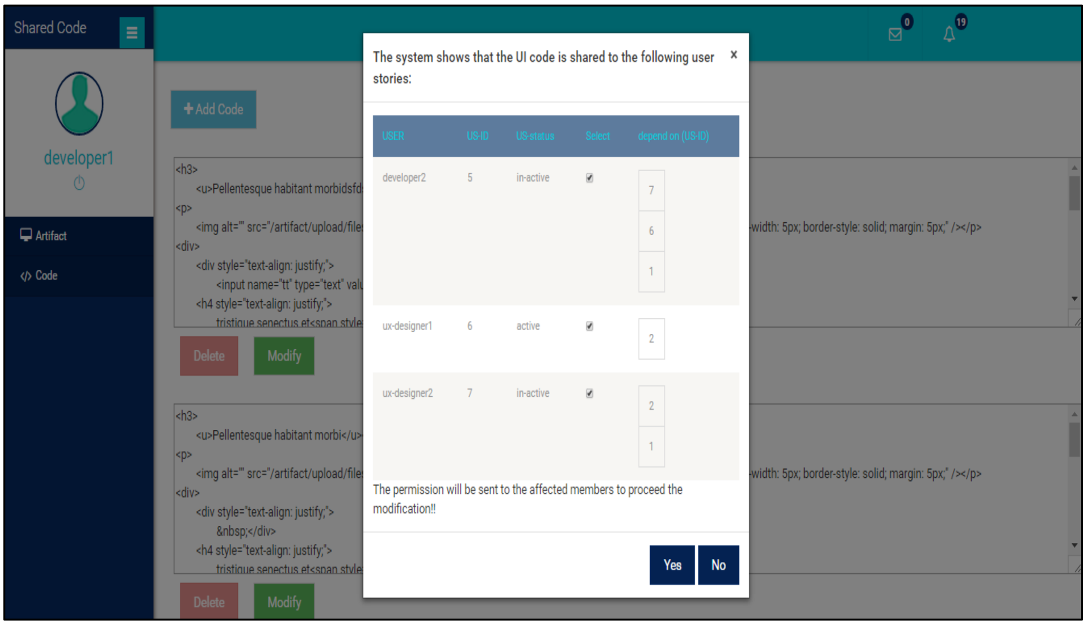


```
<h3>
    <u>Pellentesque habitant morbidsfdsfsdf</u></h3>
<p>
    <img alt="" src="/artifact/upload/files/sample.jpg" style="text-align: justify; width: 200px; height: 133px; float: right; border-width: 5px; bor
<div>
    <div style="text-align: justify;">
        <input name="tt" type="text" value="jljl" /></div>
    <h4 style="text-align: justify;">
        tristique senectus et<span style="color:#a52a2a;"> netus et malesuada fames ac </span></h4>
```

```
<h3>
    <u>Pellentesque habitant morbi</u></h3>
<p>
    <img alt="" src="/artifact/upload/files/sample.jpg" style="text-align: justify; width: 200px; height: 133px; float: right; border-width: 5px; bor
<div>
    <div style="text-align: justify;">
        &nbsp;</div>
    <h4 style="text-align: justify;">
        tristique senectus et<span style="color:#a52a2a;"> netus et malesuada fames ac </span></h4>
```

Figure 1 The implementation of scenario 1 step1

2. Developer1 will be alerted about the affected user stories (US5, US6, and US7) as potentially affected stories, along with the status for each one. Because at least one of the related user stories is 'in-active', the system informs developer1 that an approval is needed from the affected team members (user story owner: 'developer2', UX-designer: 'UX-designer1'and 'UX-designer2') before proceeding to modify codeId135 which relates to 'in-active' user stories US5 and US7, as shown in Figure 2.



The system shows that the UI code is shared to the following user stories:

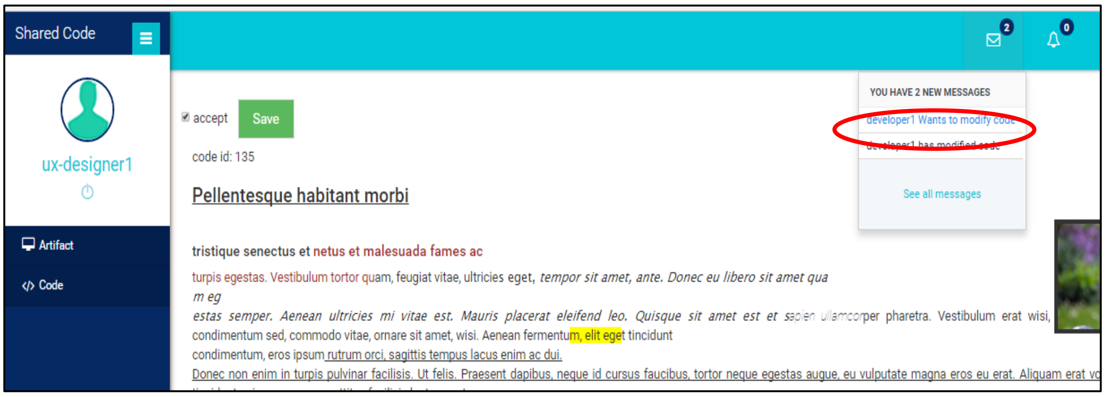
USER	US-ID	US-status	Select	depend on (US-ID)
developer2	5	in-active	☑	7 6 1
ux-designer1	6	active	☑	2
ux-designer2	7	in-active	☑	2 1

The permission will be sent to the affected members to proceed the modification!!

Yes No

Figure 2. The implementation of scenario 1 step2

3. The request will be sent to developer2, UX-designer1 and UX-designer2, as shown in Figure 3.



☑ accept Save

code id: 135

YOU HAVE 2 NEW MESSAGES

developer1 Wants to modify code

developer1 has modified code

See all messages

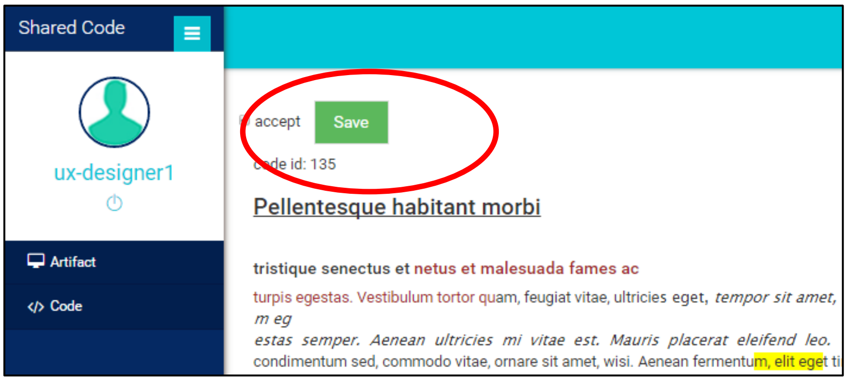
Pellentesque habitant morbi

tristique senectus et netus et malesuada fames ac

turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, *tempor sit amet, ante. Donec eu libero sit amet qua m eg estas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et sapien ullamcor*per pharetra. Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget tincidunt condimentum, eros ipsum rutrum orci, sagittis tempus lacus enim ac dui. Donec non enim in turpis pulvinar facilisis. Ut felis. Praesent dapibus, neque id cursus faucibus, tortor neque egestas augue, eu vulputate magna eros eu erat. Aliquam erat vo

Figure 3. The implementation of scenario 1 step3

4. When the request is accepted by the affected member, developer1 will be notified and the user story status be changed to ‘active’, as shown in Figure 4.



code id: 135

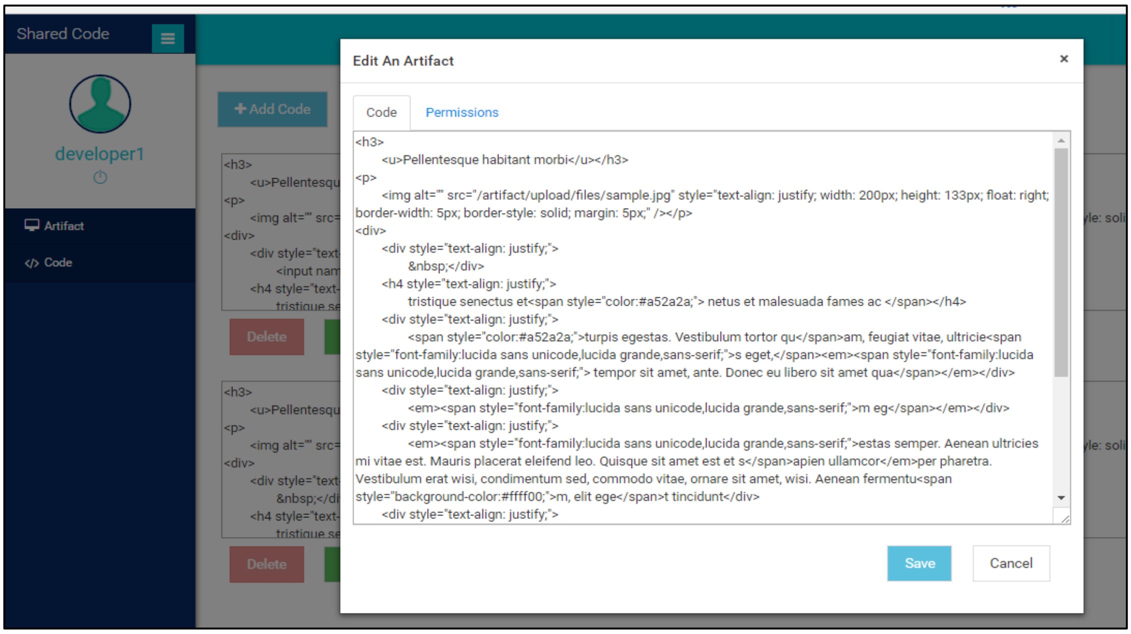
Pellentesque habitant morbi

tristique senectus et netus et malesuada fames ac

turpis egestas. Vestibulum tortor quam, feugiat vitae, ultricies eget, *tempor sit amet, m eg*
estas semper. Aenean ultricies mi vitae est. Mauris placerat eleifend leo.
condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentum, elit eget ti

Figure 4. The implementation of scenario1 step4

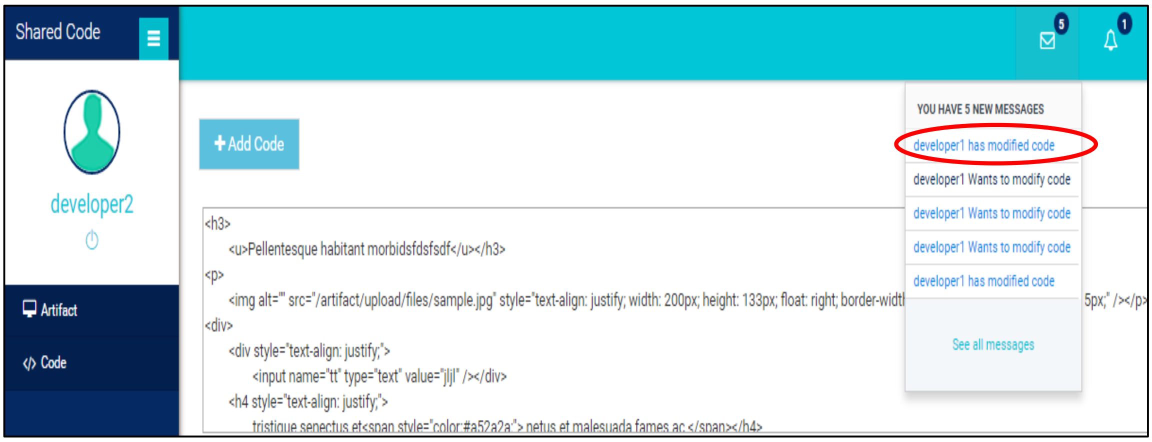
5. Modification will then be available for developer1 as shown in Figure 5.



```
<h3>
    <u>Pellentesque habitant morbi</u></h3>
<p>
    <img alt="" src="/artifact/upload/files/sample.jpg" style="text-align: justify; width: 200px; height: 133px; float: right;
border-width: 5px; border-style: solid; margin: 5px;" /></p>
<div>
    <div style="text-align: justify;">
        &nbsp;</div>
    <h4 style="text-align: justify;">
        tristique senectus et<span style="color:#a52a2a;"> netus et malesuada fames ac </span></h4>
    <div style="text-align: justify;">
        <span style="color:#a52a2a;">turpis egestas. Vestibulum tortor qu</span>am, feugiat vitae, ultricie<span
style="font-family:lucida sans unicode,lucida grande,sans-serif;">s eget,</span><em><span style="font-family:lucida
sans unicode,lucida grande,sans-serif;"> tempor sit amet, ante. Donec eu libero sit amet qua</span></em></div>
    <div style="text-align: justify;">
        <em><span style="font-family:lucida sans unicode,lucida grande,sans-serif;">m eg</span></em></div>
    <div style="text-align: justify;">
        <em><span style="font-family:lucida sans unicode,lucida grande,sans-serif;">estas semper. Aenean ultricies
mi vitae est. Mauris placerat eleifend leo. Quisque sit amet est et s</span>apien ullamcor</em>per pharetra.
Vestibulum erat wisi, condimentum sed, commodo vitae, ornare sit amet, wisi. Aenean fermentu<span
style="background-color:#ffff00;">m, elit ege</span>t tincidunt</div>
    <div style="text-align: justify;">
```

Figure 5. The implementation of scenario1 step5

6. All other affected members will be aware once the modification has been applied, as shown in Figure 6.



YOU HAVE 5 NEW MESSAGES

developer1 has modified code

developer1 Wants to modify code

developer1 Wants to modify code

developer1 Wants to modify code

developer1 has modified code

```
<h3>
    <u>Pellentesque habitant morbidsfdsfsdf</u></h3>
<p>
    <img alt="" src="/artifact/upload/files/sample.jpg" style="text-align: justify; width: 200px; height: 133px; float: right; border-widtl          5px;" /></p>
<div>
    <div style="text-align: justify;">
        <input name="tt" type="text" value="jljl" /></div>
    <h4 style="text-align: justify;">
        tristique senectus et<span style="color:#a52a2a;"> netus et malesuada fames ac </span></h4>
```

Figure 6- The implementation of scenario1 step6

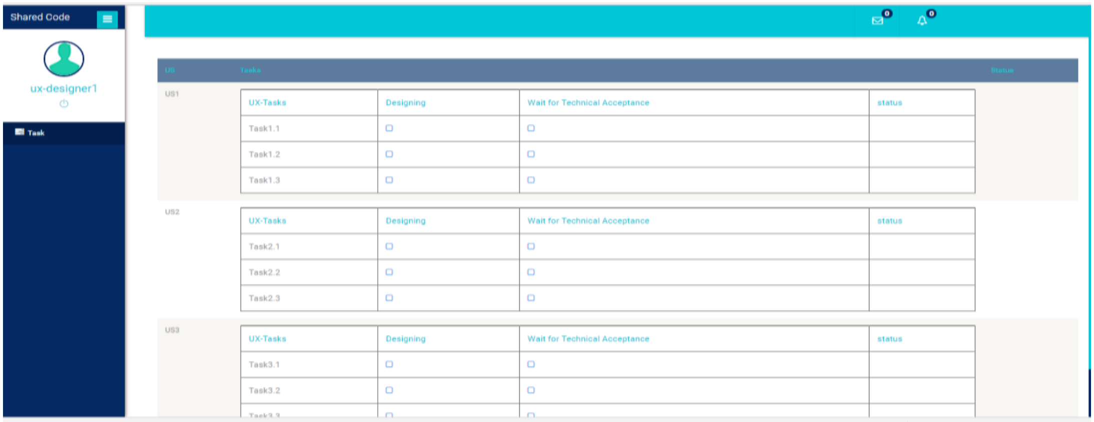
Scenarios 2: Managing the user story that contains the UX-task being designed

Scenario 2 Description

1. UX-designer1 identifies ‘Task2.1’.
2. The system checks the related user stories which is US2 owned by developer1.
3. The system checks the status of US2 which is ‘in-active’.
4. UX-designer1 informed to send a request to developer1.
5. Developer1 changes US2 to be ‘active’
6. UX-designer1 starts designing ‘Task2.1’.
7. The status of ‘Task2.1’ is updated to ‘being designed’ in the design backlog. The system changes status of the dependent US2 as (waiting for design) in the product backlog.

Running Scenario 2

In scenario 2 if the role is (UX-designer) then he is supposed to view this (design backlog) as shown on Figure 7:



US1

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task1.1	□	□	
Task1.2	□	□	
Task1.3	□	□	

US2

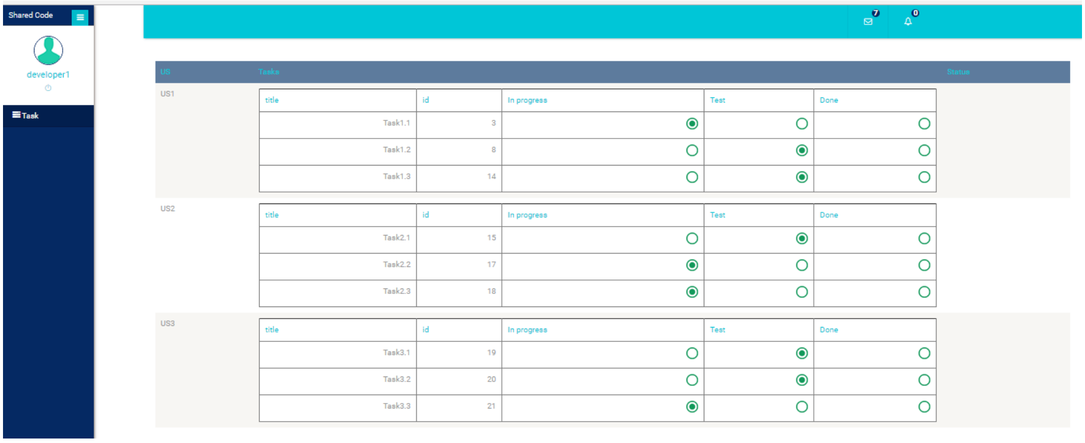
UX-Tasks	Designing	Wait for Technical Acceptance	status
Task2.1	□	□	
Task2.2	□	□	
Task2.3	□	□	

US3

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task3.1	□	□	
Task3.2	□	□	
Task3.3	□	□	

Figure 7. Task board view for the UX-designers

If the role is (developer), then he is supposed to view this (product backlog) as shown on Figure 8.



US1

title	id	In progress	Test	Done
Task1.1	3	●	○	○
Task1.2	8	○	●	○
Task1.3	14	○	●	○

US2

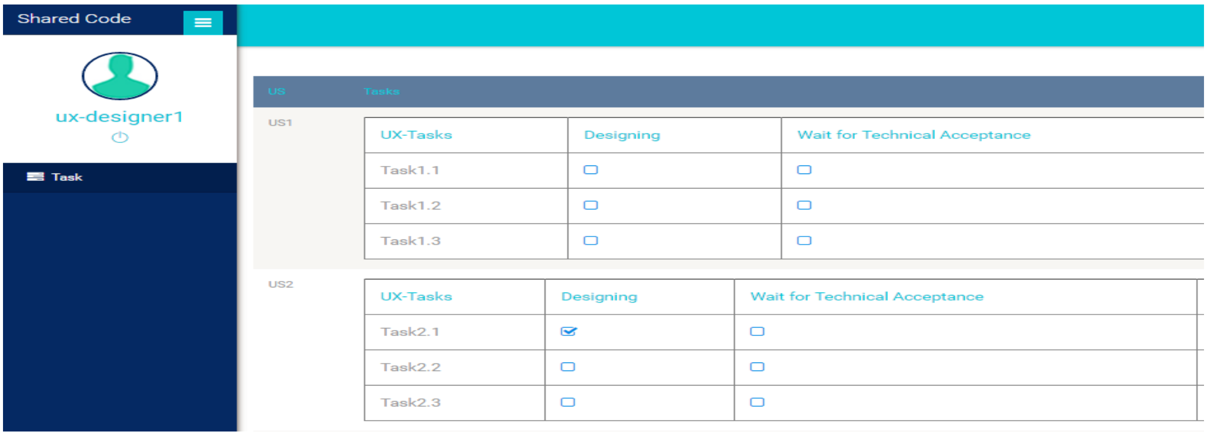
title	id	In progress	Test	Done
Task2.1	15	○	●	○
Task2.2	17	●	○	○
Task2.3	18	●	○	○

US3

title	id	In progress	Test	Done
Task3.1	19	○	●	○
Task3.2	20	○	●	○
Task3.3	21	●	○	○

Figure 8. Task board view for the developers

1. UX-designer1 identifies UX-task Task2.1 of the user story US2 as shown on Figure 9.



US1

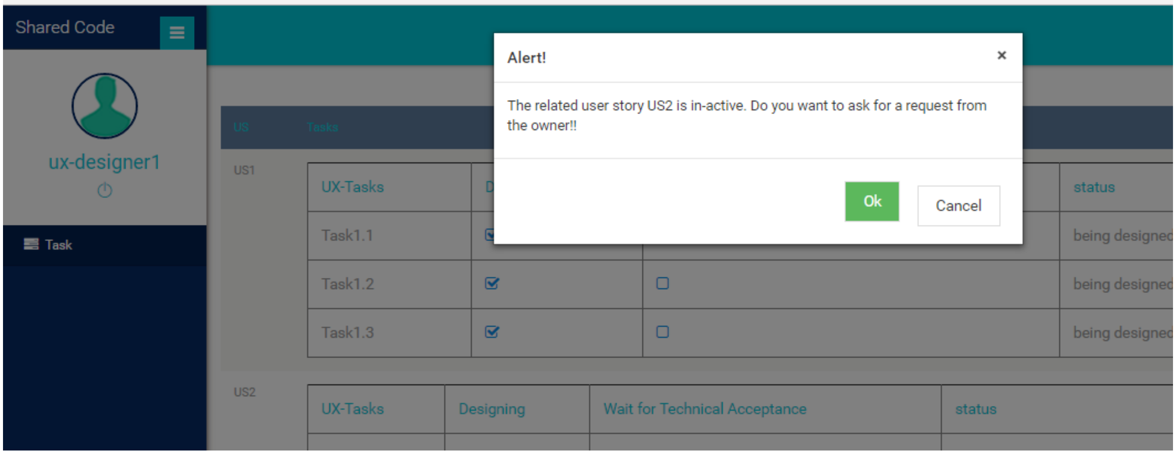
UX-Tasks	Designing	Wait for Technical Acceptance
Task1.1	□	□
Task1.2	□	□
Task1.3	□	□

US2

UX-Tasks	Designing	Wait for Technical Acceptance
Task2.1	☑	□
Task2.2	□	□
Task2.3	□	□

Figure 9. The implementation of scenario2 step1

2. The tool checks the status of US2 and related user stories, if it is in-active, the request will be sent to the owner as shown on Figure 10.



The related user story US2 is in-active. Do you want to ask for a request from the owner!!

US1

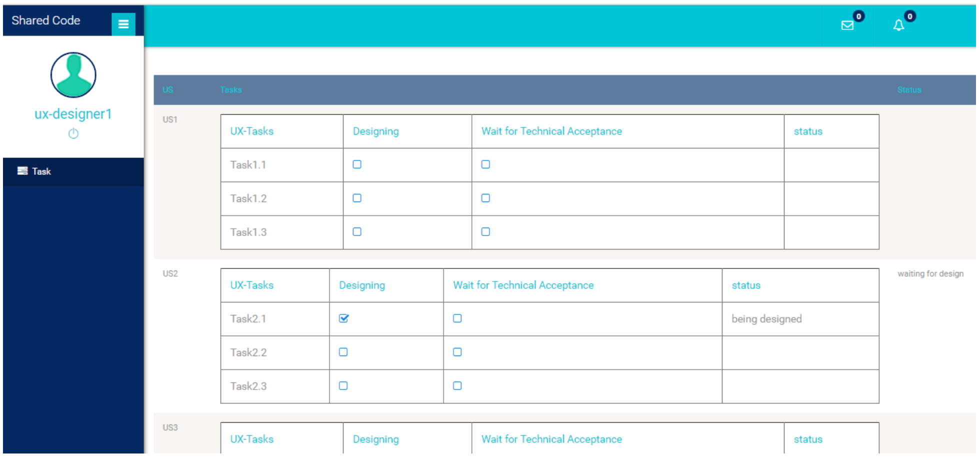
UX-Tasks	D		status
Task1.1	☑		being designed
Task1.2	☑	□	being designed
Task1.3	☑	□	being designed

US2

UX-Tasks	Designing	Wait for Technical Acceptance	status

Figure 10. The implementation of scenario2 step2

3. After accepting the request and changing the user story to active status, UX-designer1 starts designing the UX-task. The task status updated to (being designed) which implies, checking the dependent US2 and update the user story status (waiting for design) as shown on Figure 11.



US1

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task1.1	□	□	
Task1.2	□	□	
Task1.3	□	□	

US2 — waiting for design

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task2.1	☑	□	being designed
Task2.2	□	□	
Task2.3	□	□	

US3

UX-Tasks	Designing	Wait for Technical Acceptance	status

Figure 11. The implementation of scenario2 step3

Scenario 3: Accept/Reject UX-design from Technical Perspective

Scenario 3 Description

1. UX-designer1 marks Task2.1, Task2.2, and Task2.3 as 'waiting for technical acceptance'.

2. The system checks for available developers and sends a request to check the UX tasks.

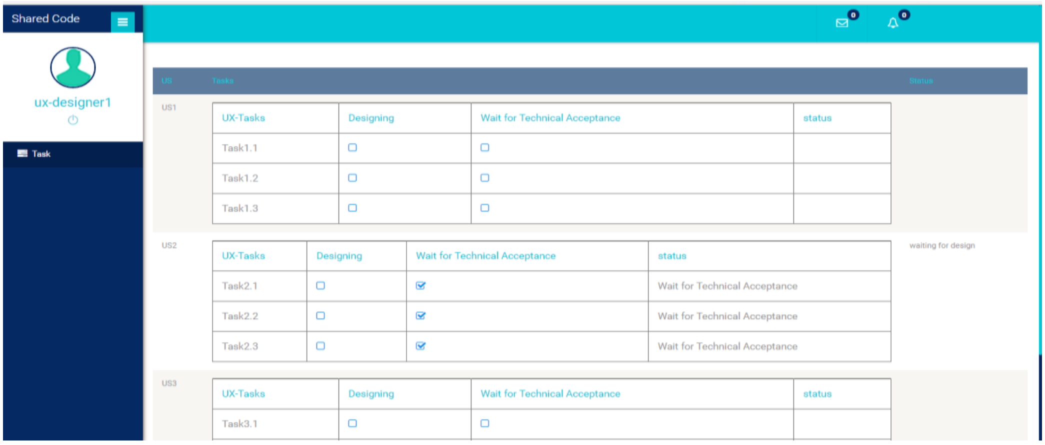
3. Developer1 accepts all the UX designs for Task2.1, Task2.2, and Task2.3.

4. The UX tasks are ‘accepted’, so the user story US2 will be automatically labeled as ‘ready for development’.

5. UX-designer1 of Task2.1, Task2.2, and Task2.3 will be notified of the acceptance.

Running Scenarios 3

1. UX-designer1 marks Task2.1, Task2.2 and Task2.3 as ‘waiting for technical acceptance’ as shown on Figure 12.



US1

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task1.1	□	□	
Task1.2	□	□	
Task1.3	□	□	

US2 — waiting for design

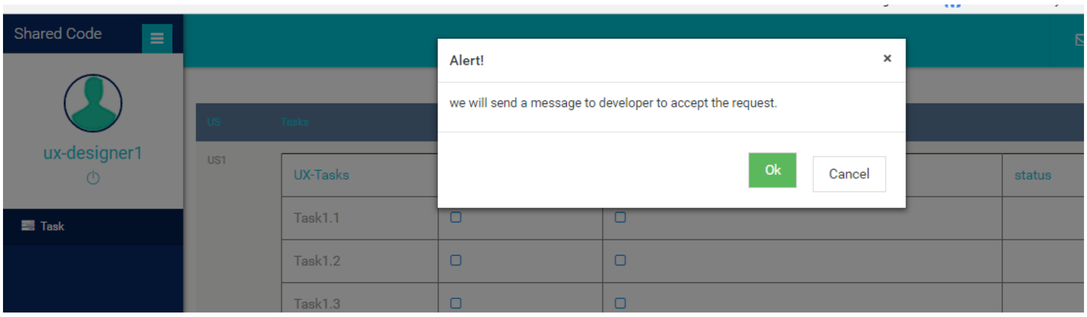
UX-Tasks	Designing	Wait for Technical Acceptance	status
Task2.1	□	☑	Wait for Technical Acceptance
Task2.2	□	☑	Wait for Technical Acceptance
Task2.3	□	☑	Wait for Technical Acceptance

US3

UX-Tasks	Designing	Wait for Technical Acceptance	status
Task3.1	□	□	

Figure 12. The implementation of scenario3 step1

2. The tool checks for available developer and sends a request to check the UX-tasks as shown on Figure 13:

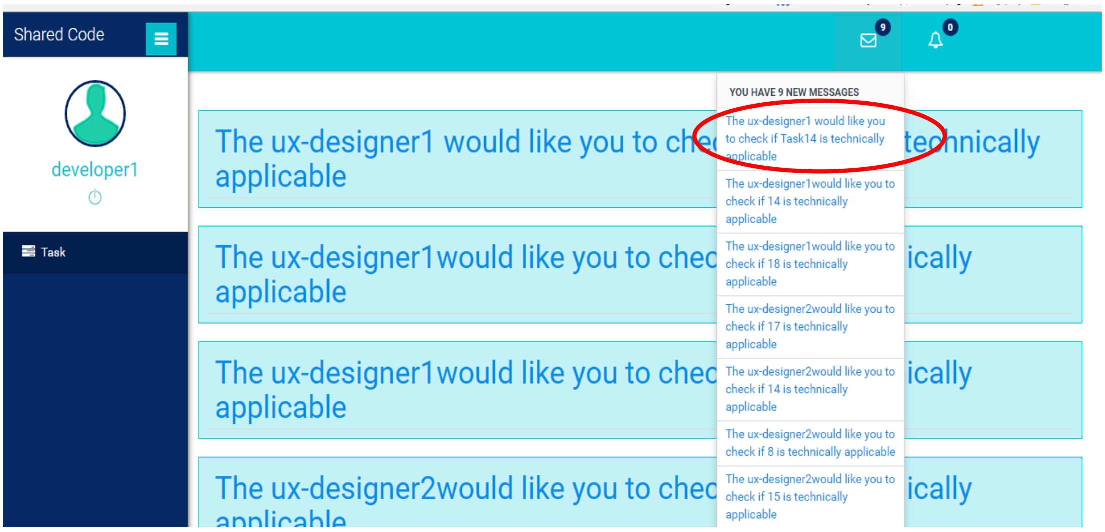


Alert!

we will send a message to developer to accept the request.

Figure 13. The implementation of scenario3 step2.

3. Developer1 accepts all the UX-designs of the UX-tasks (Task2.1, Task2.2 and Task2.3) as shown on Figure 14 and15.



YOU HAVE 9 NEW MESSAGES

- The ux-designer1 would like you to check if Task14 is technically applicable
- The ux-designer1would like you to check if 14 is technically applicable
- The ux-designer1would like you to check if 18 is technically applicable
- The ux-designer2would like you to check if 17 is technically applicable
- The ux-designer2would like you to check if 14 is technically applicable
- The ux-designer2would like you to check if 8 is technically applicable
- The ux-designer2would like you to check if 15 is technically applicable

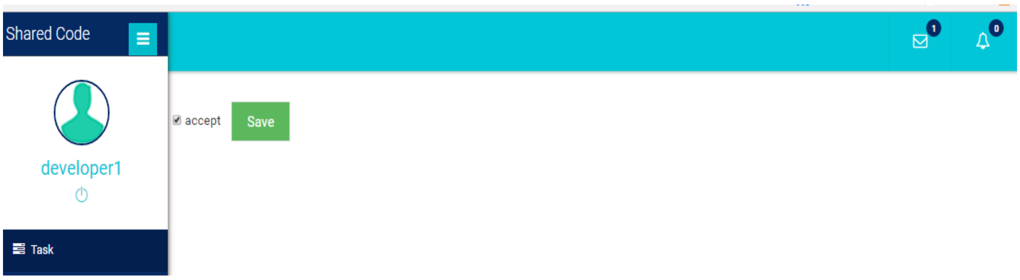
The ux-designer1 would like you to che technically applicable

The ux-designer1would like you to chec ically applicable

The ux-designer1would like you to chec ically applicable

The ux-designer2would like you to chec ically applicable

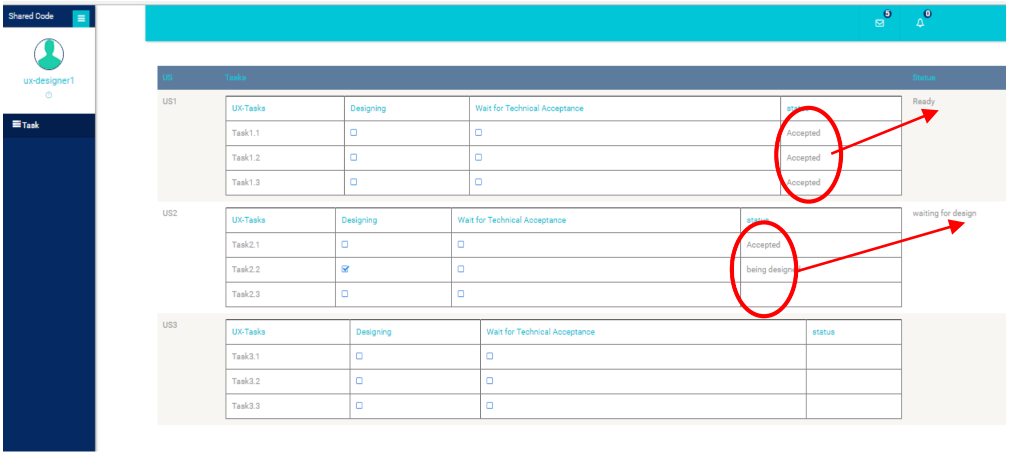
Figure 14. The implementation of scenario3 step3



☑ accept Save

Figure 15. The implementation of scenario3 step3

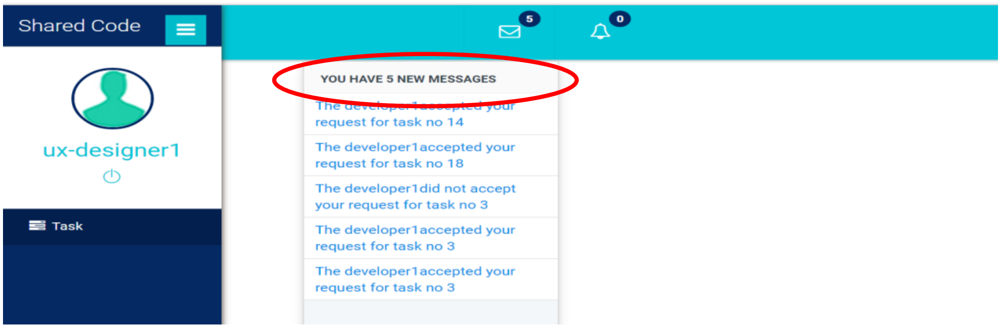
4. The tool labels US2 as ‘ready for development’; only for the user story with all UX-tasks that pass technical acceptance check as shown on Figure 16.



US	Tasks				Status
US1	UX-Tasks	Designing	Wait for Technical Acceptance	status	Ready
	Task1.1	□	□	Accepted	
	Task1.2	□	□	Accepted	
	Task1.3	□	□	Accepted	
US2	UX-Tasks	Designing	Wait for Technical Acceptance	status	waiting for design
	Task2.1	□	□	Accepted	
	Task2.2	☑	□	being design	
	Task2.3	□	□		
US3	UX-Tasks	Designing	Wait for Technical Acceptance	status	
	Task3.1	□	□		
	Task3.2	□	□		
	Task3.3	□	□		

Figure 16. The implementation of scenario3 step4.

5. The UX-designers of (Task2.1, Task2.2 and Task2.3) will be notified by the acceptance as shown on Figure 17.



YOU HAVE 5 NEW MESSAGES

The developer1accepted your request for task no 14

The developer1accepted your request for task no 18

The developer1did not accept your request for task no 3

The developer1accepted your request for task no 3

The developer1accepted your request for task no 3

Figure 17. The implementation of scenario3 step5