AI4Agile Technical Specifications and Requirements

CPTS 421 - Bolong Zeng
By: Phong Bach, Emily Cawlfield, Nain Galvan, and Aric Monary
Mentor: Skip Baccus
Date Submitted: 10/30/20

1. Updated Project Description

The project description created to explain and scope the project has evolved in the following ways:

- The subcomponents of the app have moved away from the initial four engine architecture proposed in the research paper the project is based on [5] to three processes with an additional visualization component.
- The timeline and the Gantt Chart for the project has shifted to reflect new deliverables and scheduling conflicts.
- Additional explanation about limitations associated with training data and user inputted epics.

Link: Updated Project Description

2. Client and Stakeholder Requirements and Needs

The needs and requirements of this project's clients and stakeholders are laid out in Tables 1 and 2, as well as the storyboard section beneath. For more specific metrics of these requirements, see Section 3 of this document.

2.1 Functional Requirements

ID	Description
FR1	Recommend user stories that are decomposed from an epic.
FR2	Recommend smaller user stories are split from bigger user stories.
FR3	Recommend subtasks derived from a user story.
FR4	Generate a visualization of epics, user stories, and tasks, and their relationships.

Table 1: Functional requirements.

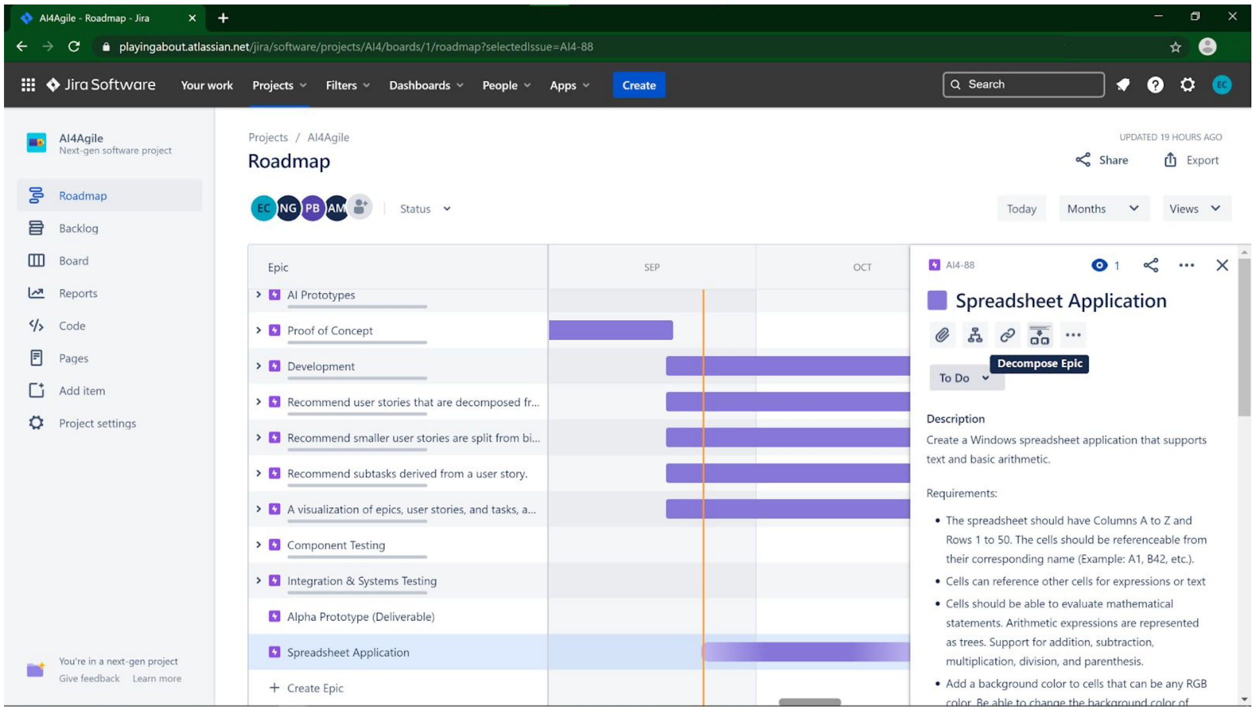
2.2 Non-functional Requirements

ID	Description
NFR1	Decompose epics and user stories without loss of information.
NFR2	Depict only necessary relationships between epics, user stories, and tasks.
NFR3	Keep all, potentially confidential, project information secure.
NFR4	Usable for users without domain expertise
NFR5	Usable by users both familiar and unfamiliar with agile development
NFR6	Extensible to different platforms
NFR7	Fast response time for AI processing

Table 2: Non-functional requirements.

2.3 Storyboard

2.3.1 Scenario 1


AI4Agile - Roadmap - Jira
playingabout.atlassian.net/jira/software/projects/AI4/boards/1/roadmap?selectedIssue=AI4-88
Jira Software
Your work
Projects
Filters
Dashboards
People
Apps
Create
Search
AI4Agile
Next-gen software project
Roadmap
Backlog
Board
Reports
Code
Pages
Add item
Project settings
You're in a next-gen project
Give feedback
Learn more
Projects / AI4Agile
Roadmap
UPDATED 19 HOURS AGO
Share
Export
EC
NG
PB
AM
Status
Today
Months
Views
Epic
SEP
OCT
AI Prototypes
Proof of Concept
Development
Recommend user stories that are decomposed fr...
Recommend smaller user stories are split from bi...
Recommend subtasks derived from a user story.
A visualization of epics, user stories, and tasks, a...
Component Testing
Integration & Systems Testing
Alpha Prototype (Deliverable)
Spreadsheet Application
Create Epic
AI4-88
Spreadsheet Application
Decompose Epic
To Do

Description

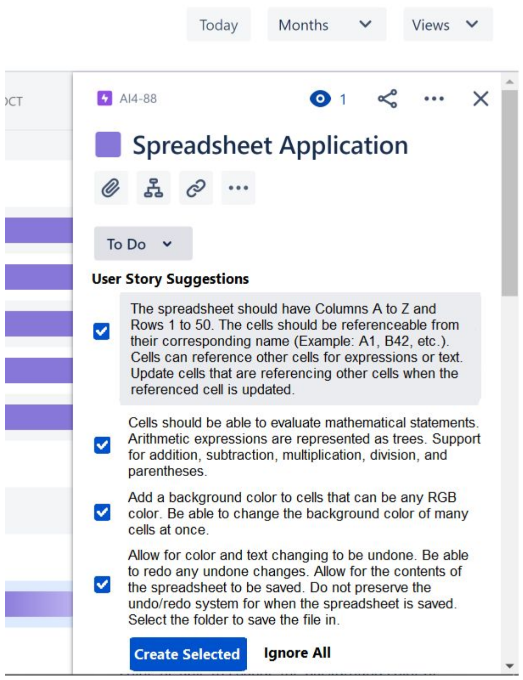
Create a Windows spreadsheet application that supports text and basic arithmetic.

Requirements:

- The spreadsheet should have Columns A to Z and Rows 1 to 50. The cells should be referenceable from their corresponding name (Example: A1, B42, etc.).
- Cells can reference other cells for expressions or text
- Cells should be able to evaluate mathematical statements. Arithmetic expressions are represented as trees. Support for addition, subtraction, multiplication, division, and parenthesis.
- Add a background color to cells that can be any RGB color. Be able to change the background color of

Figure 1: Overall view of Jira Roadmap with Decompose Epic button in focus.

Frank is a software requirements analyst, and he wants to speed up the process of breaking an epic full of requirements into smaller user stories. For this purpose, he installs the AI4Agile plugin to Jira. Frank creates a new epic, puts the requirements in with the format specified in the plugin's user manual, and clicks the *Decompose Epic* button.



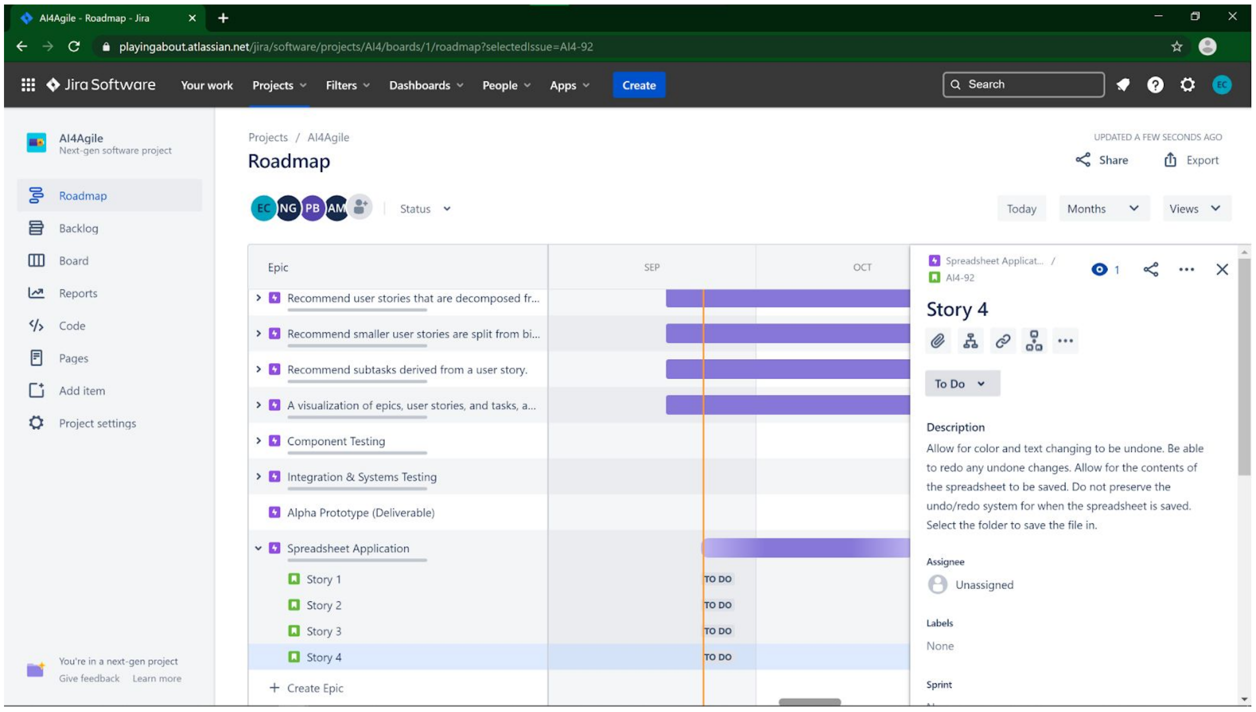
Spreadsheet Application

User Story Suggestions

- [✓] The spreadsheet should have Columns A to Z and Rows 1 to 50. The cells should be referenceable from their corresponding name (Example: A1, B42, etc.). Cells can reference other cells for expressions or text. Update cells that are referencing other cells when the referenced cell is updated.
- [✓] Cells should be able to evaluate mathematical statements. Arithmetic expressions are represented as trees. Support for addition, subtraction, multiplication, division, and parentheses.
- [✓] Add a background color to cells that can be any RGB color. Be able to change the background color of many cells at once.
- [✓] Allow for color and text changing to be undone. Be able to redo any undone changes. Allow for the contents of the spreadsheet to be saved. Do not preserve the undo/redo system for when the spreadsheet is saved. Select the folder to save the file in.

Figure 2: View of generated User Story Suggestions

Now, Frank looks at the suggestions the AI came up with. If he decides he doesn't like any of these suggestions, he can click the *Ignore All* button to go back to his Spreadsheet Application epic. To edit a story, he can click on its text box, then save or cancel those edits. To ignore a story suggestion entirely, he can uncheck its box. Once Frank is happy with his resulting user stories, he clicks *Create Selected*.



Roadmap

Epic
Recommend user stories that are decomposed fr...
Recommend smaller user stories are split from bi...
Recommend subtasks derived from a user story.
A visualization of epics, user stories, and tasks, a...
Component Testing
Integration & Systems Testing
Alpha Prototype (Deliverable)
Spreadsheet Application
Story 1 — TO DO
Story 2 — TO DO
Story 3 — TO DO
Story 4 — TO DO

Story 4

Description

Allow for color and text changing to be undone. Be able to redo any undone changes. Allow for the contents of the spreadsheet to be saved. Do not preserve the undo/redo system for when the spreadsheet is saved. Select the folder to save the file in.

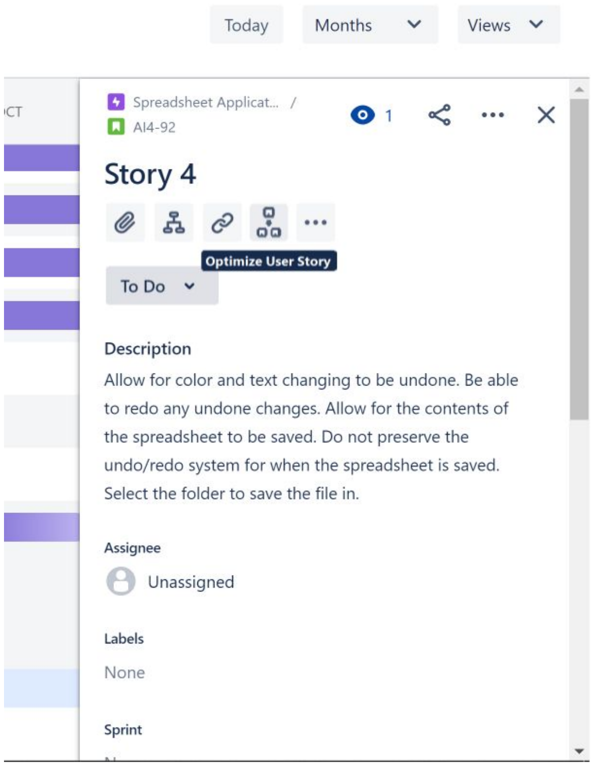
Assignee: Unassigned

Labels: None

Sprint

Figure 3: Overall view of newly created user stories.

The app has now created the 4 user stories that Frank approved under the heading of the Spreadsheet Application epic.



Description

Allow for color and text changing to be undone. Be able to redo any undone changes. Allow for the contents of the spreadsheet to be saved. Do not preserve the undo/redo system for when the spreadsheet is saved. Select the folder to save the file in.

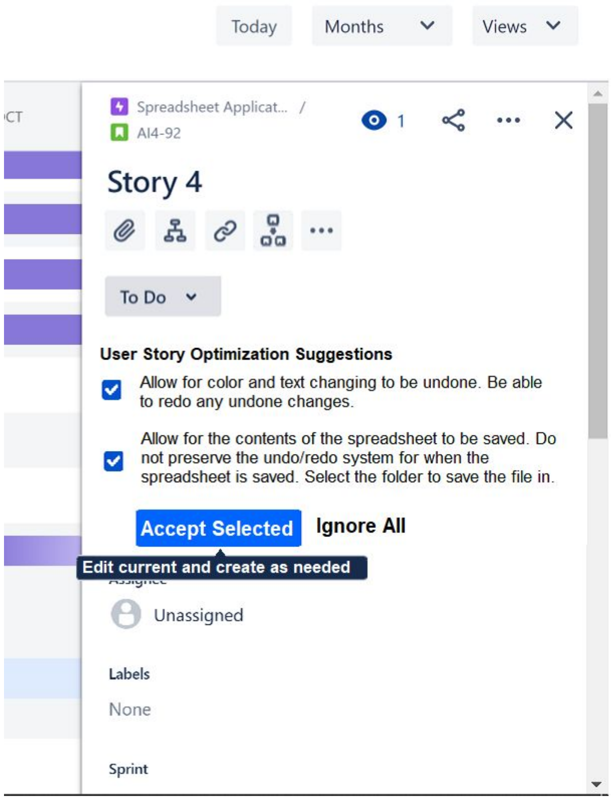
Assignee: Unassigned

Labels: None

Sprint

Figure 4: Demonstrating the Optimize User Story button.

If Frank wants to continue the process of breaking up epics, he can go into one of the user stories and click *Optimize User Story*. Depending on the size of the user story already, it might be broken down into multiple smaller stories, or left alone if it’s already optimized.



User Story Optimization Suggestions

- [x] Allow for color and text changing to be undone. Be able to redo any undone changes.
- [x] Allow for the contents of the spreadsheet to be saved. Do not preserve the undo/redo system for when the spreadsheet is saved. Select the folder to save the file in.

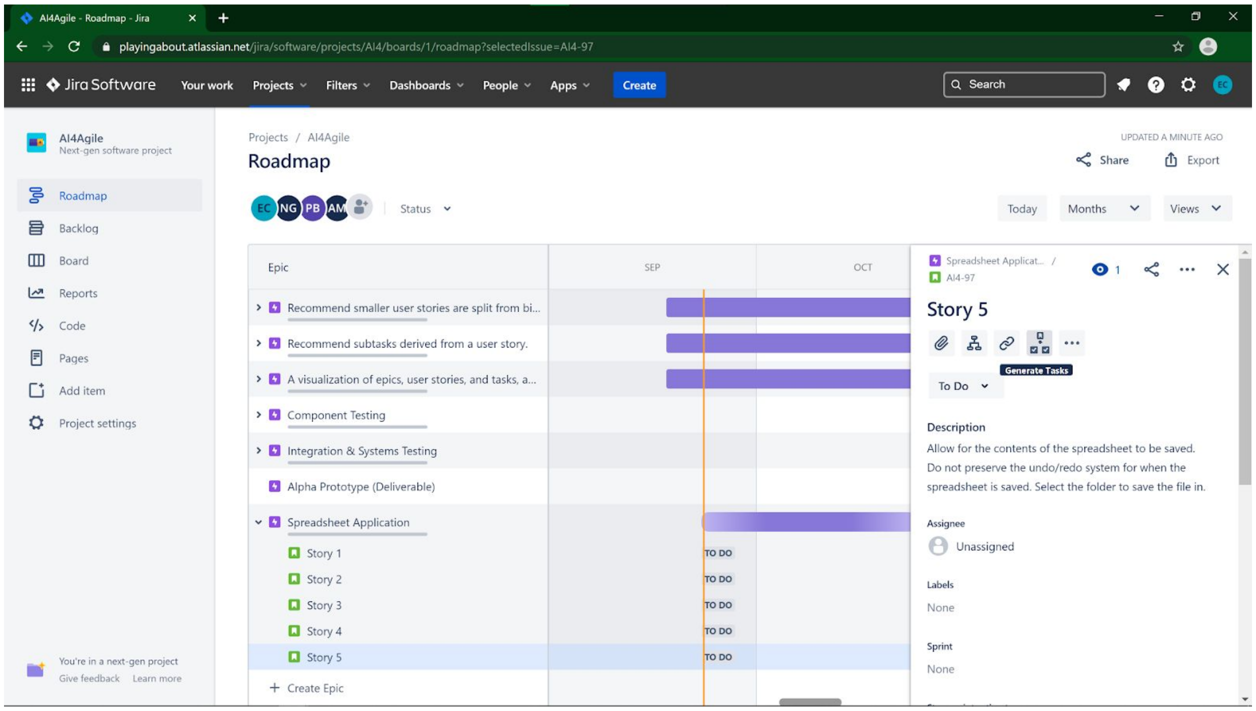
Unassigned

Labels: None

Sprint

Figure 5: User Story Optimization Suggestions page.

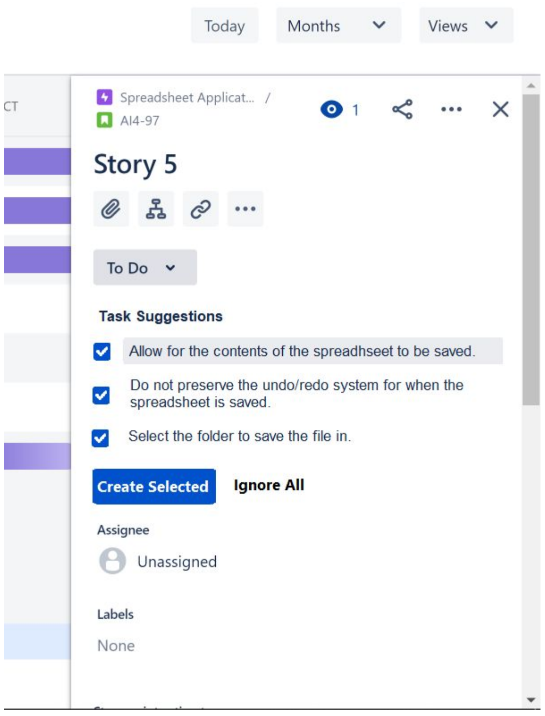
Once the Story Optimization Suggestions are generated, Frank has the same options as with the previous stories: to select or deselect stories via the checkboxes, make edits, or ignore all suggestions. As he hovers over the *Accept Selected* button, Frank is warned that this option will edit his currently selected user story as well as create a new one to accommodate for the splitting. He clicks anyway, since this is what he wanted.



Allow for the contents of the spreadsheet to be saved. Do not preserve the undo/redo system for when the spreadsheet is saved. Select the folder to save the file in.

Figure 6: View of newly created story from optimization process, and emphasis on Generate Tasks button.

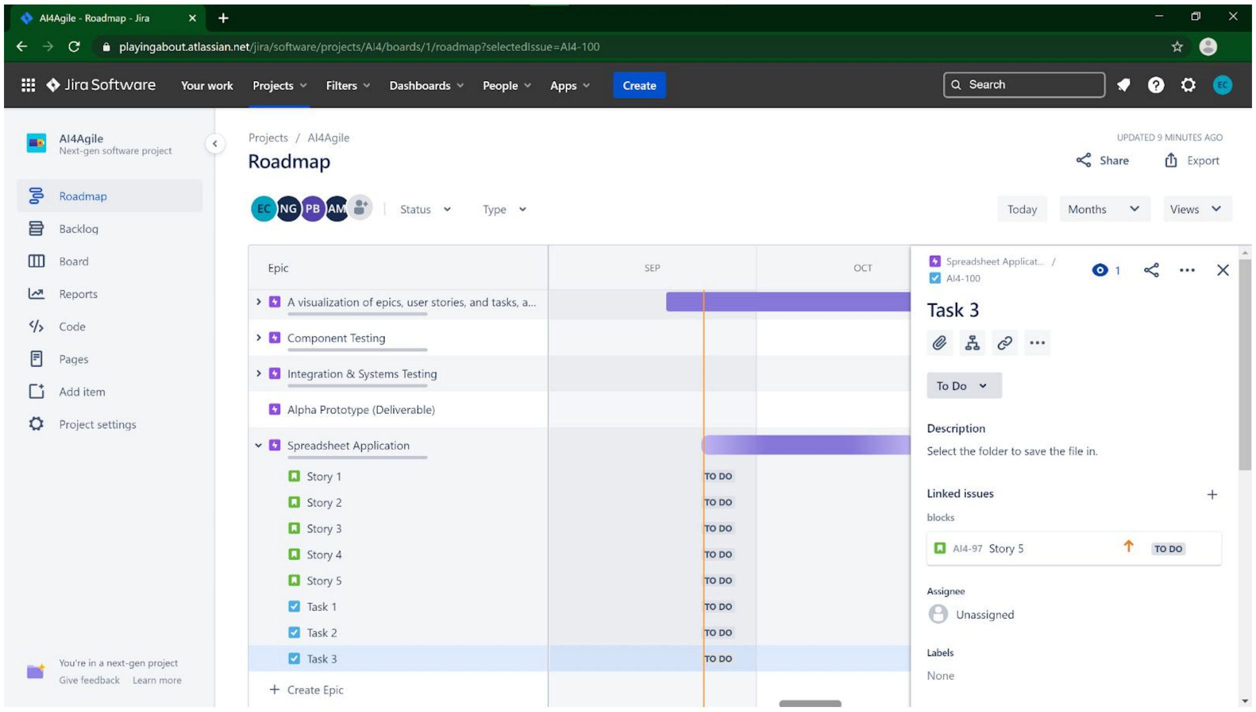
Now there are five user stories, since Story 4 was optimized into two separate stories. To continue the decomposition process, Frank opens a story and clicks *Generate Tasks*.



- [x] Allow for the contents of the spreadhseet to be saved.
- [x] Do not preserve the undo/redo system for when the spreadsheet is saved.
- [x] Select the folder to save the file in.

Figure 7: Task Suggestions page.

As before, the options include selecting or deselecting individual suggestions, editing, and ignoring all suggestions. Once he's happy with the tasks, he clicks *Create Selected*.



Epic	
A visualization of epics, user stories, and tasks, a...	
Component Testing	
Integration & Systems Testing	
Alpha Prototype (Deliverable)	
Spreadsheet Application	
Story 1	TO DO
Story 2	TO DO
Story 3	TO DO
Story 4	TO DO
Story 5	TO DO
Task 1	TO DO
Task 2	TO DO
Task 3	TO DO

Task 3

To Do

Description
Select the folder to save the file in.

Linked issues
blocks
AI4-97 Story 5 TO DO

Assignee
Unassigned

Labels
None

Figure 8: View of newly created tasks for one fully decomposed user story.

The tasks have now been created, and linked to their parent story to indicate a blocking relationship. All Frank had to do was make some decisions and maybe edits, and now he's got one story fully decomposed!

2.3.2 Scenario 2

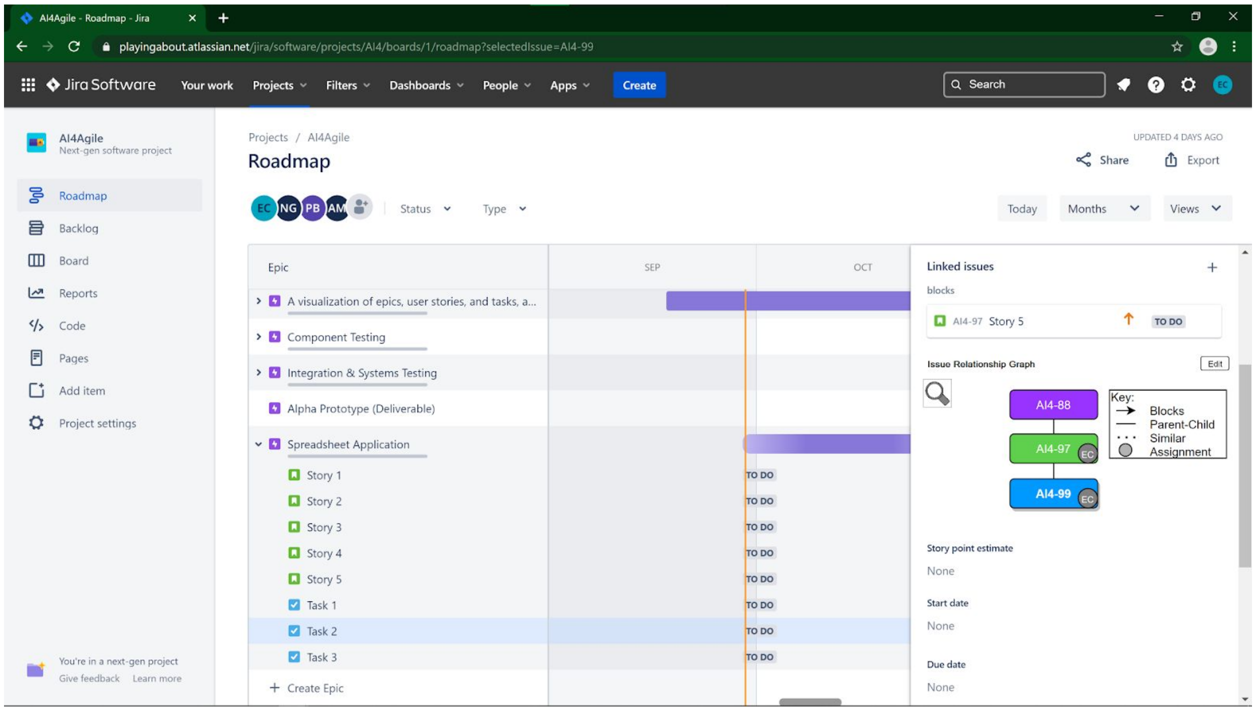
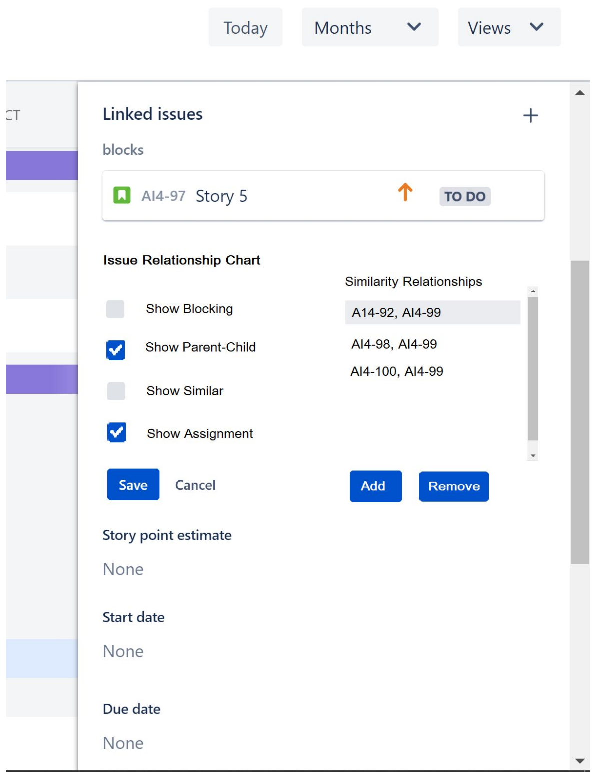


Figure 9: A zoomed out view of navigating to the issue relationship graph.

Madison is a scrum master, and she wants to take a closer look at some of the tasks to make sure the developer assignments make sense. To do this, she uses the AI4Agile Issue Relationship Graph feature by navigating to a certain epic, user story, or task, and finding the graph portion. From here, she has the options to *Zoom In* on the graph, or *Edit* it to change which types of relationships are shown. Madison wants to show more relationships, so she clicks *Edit*.



Linked issues

blocks

AI4-97 Story 5 TO DO

Issue Relationship Chart

- [ ] Show Blocking
- [x] Show Parent-Child
- [ ] Show Similar
- [x] Show Assignment

Save Cancel

Similarity Relationships

A14-92, AI4-99

AI4-98, AI4-99

AI4-100, AI4-99

Add Remove

Story point estimate

None

Start date

None

Due date

None

Figure 10: Graph editing options.

In the graph's edit page, Madison checks the boxes to *Show Blocking* and *Show Similar* so that those types of relationships will populate on the graph. If she thought any of the suggested similar issues were incorrect or lacking, she could use the menu on the right to change them. Next, she clicks to *Save* these options.

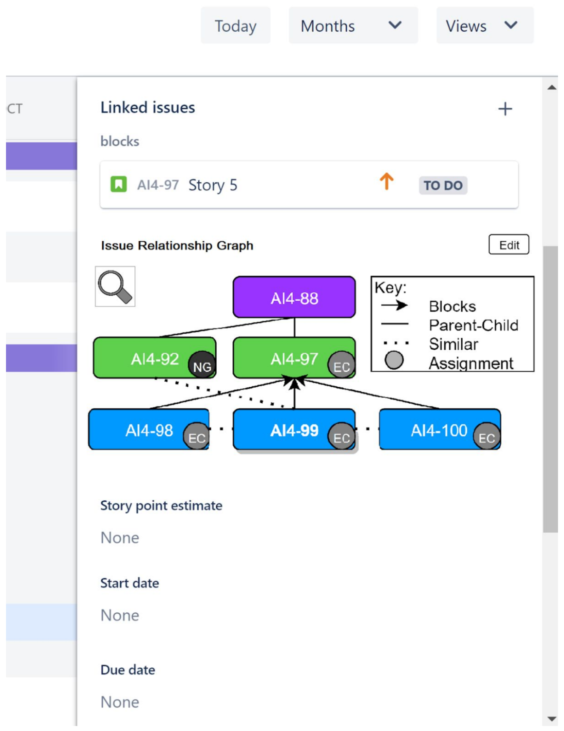


Figure 11: New relationship graph after edits.

Now, Madison can see all the available relationships between the task she has selected and other tasks, stories, and the epic they came from. She is better able to decide whether the team members she assigned to these pieces make sense given the relations between them.

3. Mapping of Requirements to Technical Specifications

3.1 Stakeholder Needs and Use Cases

No.	Needs	Metrics
1	Can decompose epic into user story	AI understands what is an epic
2	Can optimize user story into small user story	AI understands what is an user story
3	Can generate tasks from user stories	AI understand what is a task
4	Can render relationship graphs	Statistic component Graphical component
5	Is easy to use	Time to navigate the decompose
6	Support English language	NLP component
7	Allow different level of prediction	Prediction slider
8	Allow user to select or discard prediction	Options to select or discard prediction
9	Allow user to select different relationship graphs	Options to to select different relationship graph
10	Remains stable under multiple decompositions	Range of times an input can be decomposed
11	Can process a wide variety of input types	Amount of type of input that can be decomposed
12	Is responsive	Response time
13	Preserve information integrity	Percentage of information retained
14	Provide security to user data	Authentication system
15	Can regenerate different output	Amount of new output can generate
16	AI can be easily retrained	Time to train AI

Table 3: Needs and metrics table

3.2 Metric Targets

No.	Metrics	Target
1	AI understands what is an epic	Minimum 3 Clusters
2	AI understands what is an user story	Minimum 2 prediction
3	AI understands what is a task	Minimum 3 Slices
4	Statistic component	Percentage of each components
5	Graphical component	Tree and cluster charts
6	Time to navigate the decompose	Maximum 2 minutes
7	NLP component	Word2Vec, Doc2Vec
8	Prediction slider	Minimum 2 options level
9	Options to select or discard prediction	Yes, no
10	Options to to select different relationship graph	Epic, user story, task
11	Bounds of time to decompose an input	From 1 - 5 seconds
12	Variety of types of input that can be decomposed	Paragraph, sentence
13	Response time	From 1 - 2 seconds
14	Percentage of information retained	From 90% - 100%
15	Authentication system	Minimum 1 type of encryption
16	Amount of new output AI can generate	Minimum 1 output
17	Time to update and retrain AI	Maximum

Table 4: Technical target specification table

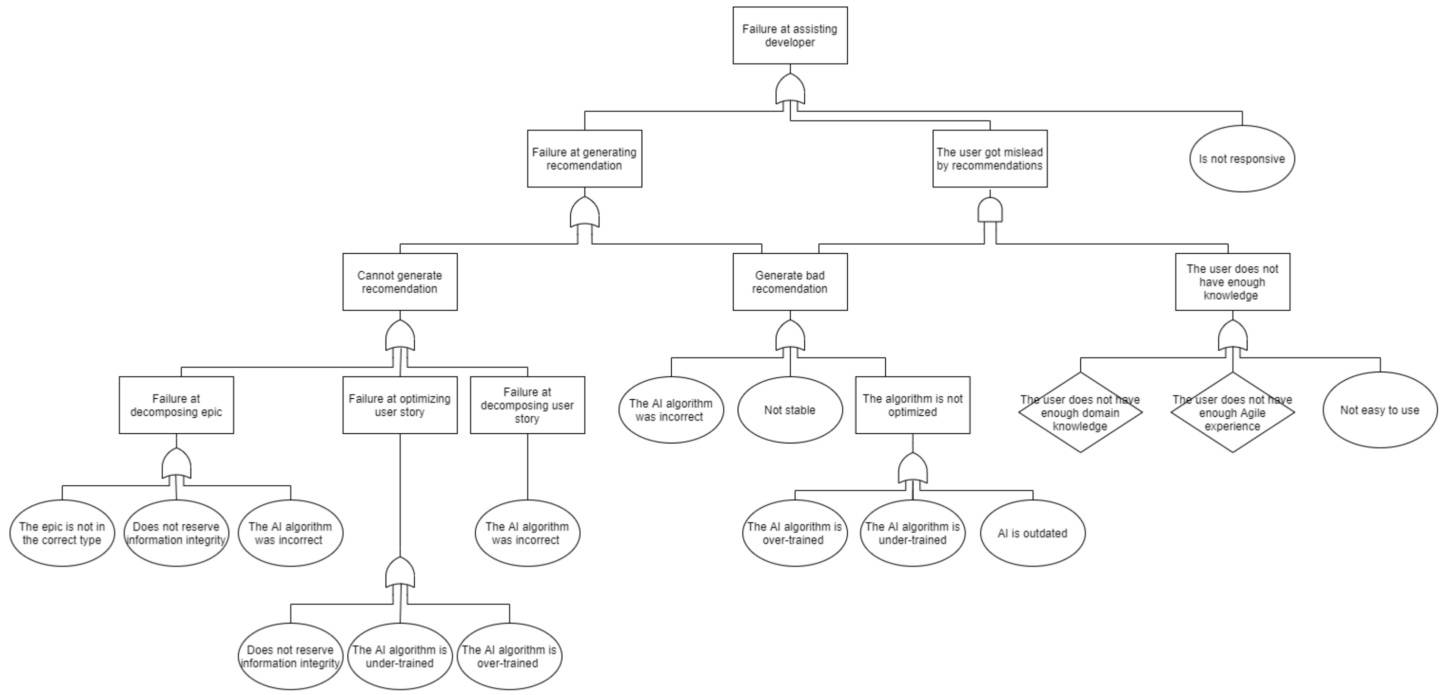

Failure at assisting developer
Failure at generating recomendation
The user got mislead by recommendations
Is not responsive
Cannot generate recomendation
Generate bad recomendation
The user does not have enough knowledge
Failure at decomposing epic
Failure at optimizing user story
Failure at decomposing user story
The AI algorithm was incorrect
Not stable
The algorithm is not optimized
The user does not have enough domain knowledge
The user does not have enough Agile experience
Not easy to use
The epic is not in the correct type
Does not reserve information integrity
The AI algorithm was incorrect
The AI algorithm was incorrect
The AI algorithm is over-trained
The AI algorithm is under-trained
AI is outdated
Does not reserve information integrity
The AI algorithm is under-trained
The AI algorithm is over-trained

Figure 12: Fault tree diagram for AI4Agile

The impacts on the user can be analyzed using Use Case Scenarios technique to map all needs to target technical specification. From the four main use cases, recommending user story that are decomposed from an epic, recommending subtasks derived from a user story, recommending smaller user stories are split from bigger user stories, and visualization of epics, user stories and tasks, and their relationship, the corresponding needs and metrics can be derived such that the AI understands what is an epic, what is an user story, and what is a task. From other use cases in Table 3, the table needs and metrics was developed to prepare the metrics for establishing target specifications. From Table 3, the technical target specification, found in Table 4, was generated for all the metrics in the project.

Some of the broader impacts are potentially misleading developers with limited domain knowledge and the project is delayed from our bad suggestion. From Figure 12, it shows that from the misled of the user by AI’s recommendation, it can be caused by the AI giving unrelated recommendations and the user not having enough knowledge. The cause of the AI giving unrelated recommendations can be either that the AI algorithm is incorrect, under-trained, or over-trained. From these causes, the software needs to choose the correct AI algorithm for each decomposition, and the training for AI should be balanced.

In the case that the user does not have enough knowledge on the project, the problem can be either that the user does not have enough domain knowledge on the system or the user does not have enough experience in Agile development. Both of these issues can only be fixed by the user to educate themselves on the system domain knowledge and familiarize themselves with Agile development.

4. Preliminary Implementation Plan

The data components that are going to be used is a database which will store the epics, user-stories, and tasks since they could hold sensitive information. Jira will only be able to access the information in the database while having an account or specific login the database could be accessed by a user. There are four components to the application.

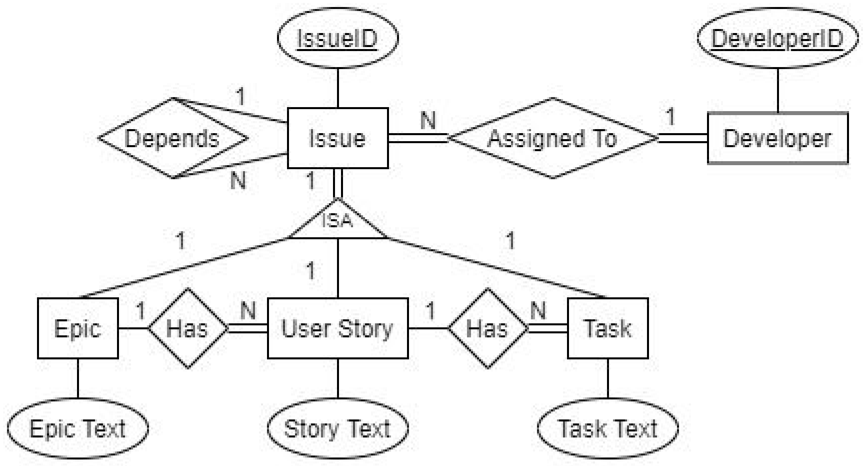


Figure 10: Entity-relationship diagram for the proposed database

The first is decomposing an epic to user-stories to accomplish this unsupervised clustering will be used. These clustering methods focus on Density-based spatial clustering, ordering points to identify clustering structure, and balanced iterative reducing and clustering using hierarchies. These methods will be analyzed by 6 different ways to determine which clusters belong alone or together. By using a template for the epic, it will help determine functional and non-functional parts to make it easier to group the clusters into user-stories.

The second is optimizing user-stories. This means that large user-stories need to be broken down into smaller ones. User-stories are general explanations of software features from the user's perspective. The main approach is using long short-term memory which determines the meaning of the word based on the previous one. This will help determine which words provide important information.

The third component is breaking user-stories into tasks. Tasks are deliverables or actions which could be done by one person in the team in one day. The best way to implement breaking user-stories into tasks is to slice the user-stories. Using NLKT library in python for NLP. This library can be used for many things such as removing stop words, tokenization, chunking and many other things. These will be used to create actionable sentences. For validation BERT and GPT will be used to compare our results with tokenization.

The last component is creating a visualization of the relationships between an epic, user-stories, and tasks. The user will be able to select a certain piece and see the relationship it belongs to. The user can select which relationships they want to be shown and they can add or remove connections. The examples of relationships that could be shown: dependencies (blocking relationships), things assigned to the same developer, and parent-child relationships. The explicit relationships will be shown as a tree, while the implicit relationships will be shown as clusters. A combination of the explicit and implicit relationships would also be shown as a tree.

References:

1. Hoa Khanh Dam, Truyen Tran, John Grundy, Aditya Ghose and Yasutaka Kamei, *Towards effective AI-powered agile project management*, Proceedings of the 41st International Conference on Software Engineering (ICSE 2019), New Ideas and Emerging Results.