ScrumHelper

Internet - Practical Course Telecooperation

Asha Joshi, 2281908
Dibyojyoti Saniyal, 2688349
Saju Daniel

TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Index

1. Introduction

Agile Methodologies is one of the modern process models to execute projects. In recent years SCRUM has become a leading agile framework for effective project management.
SCRUM as a methodology has various tools to execute the project. Namely poker cards to come up with story points. Story boards to check the regular progress to name a few. The scrum framework is designed to encourage empowerment and self organization, early and regular deliveries, inspect and adapt, time boxing and transparency.
Scrum Helper is a scrum tool to execute projects using scrum model. The tool is build using latest web technology namely MEAN stack. MEAN is a framework for an easy starting point with MongoDB, Node.js, Express, and AngularJS based applications. It is designed to give you a quick and organized way to start developing MEAN based web apps with useful modules like Mongoose
The document will provide the technical architecture of the tool and the user guidelines on how to use the tool for the execution of the project.
The project based on MEAN stack where the frontend is build using angularjs enabled webpages and the backend is supported by node.js and mongodb. This section gives a brief on the technologies used and the basic architecture of the project.

2. Technologies stack

2.1 Mean Stack

a. MongoDB

Mongodb : MongoDB is an open-source *document database* that provides high performance, high availability, and automatic scaling. MongoDB obviates the need for an Object Relational Mapping (ORM) to facilitate development. MongoDB is classified as a NoSQL database. It uses JSON like documents to dynamic schema to store data instead of the conventional table based relational database. This makes the integration of applications easier and faster.

b. Express.js

Express.js : Express.js is a Node.js web application server framework designed to build web applications .It is a standard web framework for Node.js. Express is the backend part of MEAN stack

c. Angular.js

Angular.js : AngularJS extends HTML with new attributes and makes it responsive to user actions. It is used to build Single Page Applications. It aims to simply both the development and testing of single page applications by providing a framework for client-side model-view-controller (MVC). AngularJS is the fronted part of MEAN stack.
We used following angular plugins in our website:
angular-chartjs, angular-datatables, angular-moment, angular-bootstrap-datetimepicker
We also use angular-socket-io for implementing the chat functionality where user can converse with all other users logged in at the time.

d. Node.js

Node.js : Node.js is an open-source, cross-platform runtime environment for developing server side applications. Node.js provides event driven architecture and a non-blocking I/O API which optimizes the applications throughput and scalability for real-time Web applications.

AngularJS and MongoDB both speak JSON, as do Node.js and Express.js. The data flows neatly among all the layers without rewriting or reformatting. MEAN uses the same JSON format for data everywhere, which makes it simpler and saves time reformatting as it passes through each layer. Plus, JSON's ubiquity through the MEAN stack makes working with external APIs that much easier: GET, manipulate, present, POST, and store all with one format.

2.2. Bootstrap

Bootstrap is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first web sites. We have used bootstrap which enables clean UI in all resolutions and hence can be used from Desktops, tabs to mobile phones.

2.3. Bower, Yeoman, Grunt

Bower can manage components that contain HTML, CSS, JavaScript, fonts or even image files. Bower doesn't concatenate or minify code or do anything else - it just installs the right versions of the packages you need and their dependencies. We used Bower to optimize the front-end. Bower automatically download dependent multiple packages and also filter redundant dependencies by downloading only once.

Yeoman is a generic scaffolding system allowing the creation any kind of app. It allows for rapidly getting started on new projects and streamlines the maintenance of existing projects. We used Yeoman for following.

- Creation of a project mean stack
- Creation of new sections of a project, client, server side controller, modals.
- Creation of modules or packages
- Bootstrapping new services
- Enforcing standards, best practices and style guides

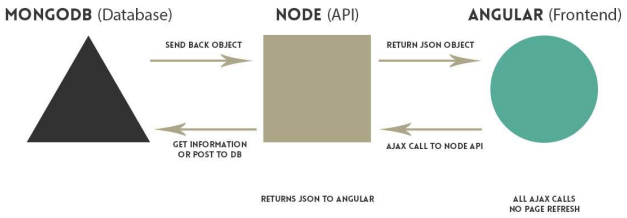
Grunt is a JavaScript task runner. It has a bunch of built-in tasks, with the ability to build your own plugins and scripts that extend the basic functionality.

2.4. Cookies

Cookies are usually small text files, given ID tags that are stored on your computer's browser directory or program data subfolders. We have used cookies to track of user project selection and activities, so that we can resume where the user left. Also we use cookies for storing temporarily all the chats (conversations) of the user.

3. Application architecture

As mentioned before the project is based on MEAN stack which provides a well defined model view controller(MVC) architecture. The application code is majorly divided into three sections as can be seen in the figure as well. The front end(angular), the backend(mongodb) and the middleware(node).

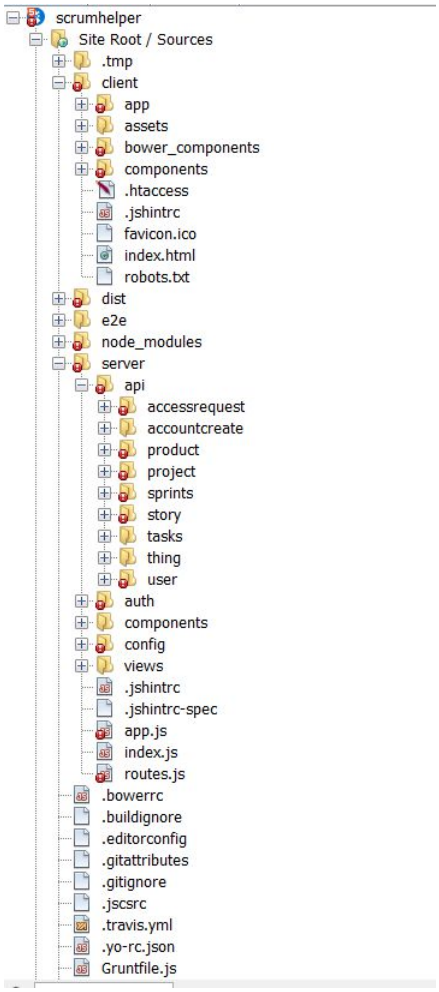


As shown the figure with code architecture, the front end code is inside client/app/ folder. Corresponding to every page there is a separate folder with corresponding html,css and controller. Similarly at the backend the code is inside server/api/ folder. Corresponding to each schema in mongodb there is a different folder, with modal and backend-controller which serves as a bridge between frontend and backend.

The components common to all the pages in front end are kept in a separate folder client/components/. These common components include the top bar, left bar, timeline and chat. These are implemented as directives and added to all webpages.

Further the chat is implemented based on web-socket service also kept inside client/components/ where as inside server/config/ at the backend. Events are forwarded from the front end to the backend from where they are broadcasted to all the clients. For example the chat application uses socket service to emit signal when user sends a message. This signal is forwarded from socket service at front end to socket service at backend. Further from backend this signal is broadcasted to all users, which is then filtered according to the intended receiver ID. This conversation is saved in cookies which is cleared once user logs out.

Furthermore we display a list of all user online and offline at the particular moment. This list must be updated whenever a new user logs in or logs out. Similar to chat , we emit signal on user login and logout which is broadcasted from the backend, and offline and online user list is updated accordingly for all users. Cookies are used to temporarily maintain this list of users until the user logs out.



In the timeline, we are displaying list of recent activities which again uses sockets to emit signals on update of creation, update and deletion of stories, sprint and task. This signals are received and broadcasted from backend then all users receiving it display the update depending on project they currently in. The activities displayed in timeline are also maintained in cookies as for chat and user login status and is cleared as soon as user logs out of application.

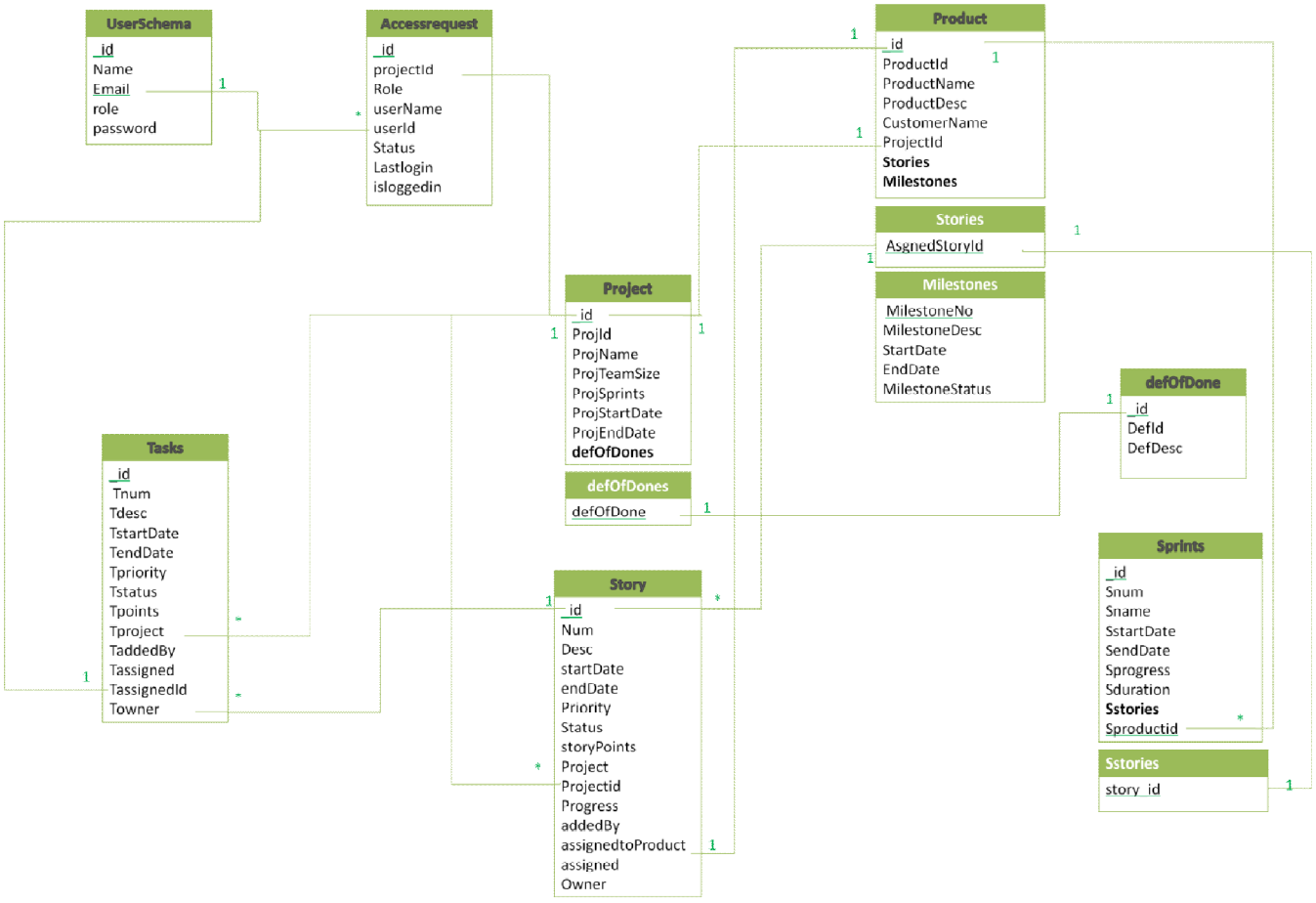
A user can be part of more than one project and products with different role. In order to display project data as
per the selected project and the options as per the role this data needs to be maintained somewhere. So for this create a service client/app/userinfo . This service keeps track of current project, product and role selected by the user. Whenever user changes the project or the role using cookies this data is retrieved and saved in the user info. When the user logs in for the first time a message is displayed to select the project and the role so that he/she can start working on scrum of that project. This data is saved and when user logs in again the view is displayed according to previously saved userinfo.
We have used a number of angular and jquery plugins to add additional features to our website and make it more user friendly and responsive. The plugin code is kept inside client/bower_components.

- Among other main plugins that we used include angular-datatable which provide a number of additional services then nomal table. Each datatable has features such as search, filtering list entries, ascending descending columns order integrated.
- angular-bootstrap makes our website responsive so that it can be used on any device with any resolution.
- Html date tags are not very user friendly and also not compatible with all browsers. For this we have used angular-bootstrap-datetimepicker which provides user friendly datepicker which is responsive and works in all browser.
- For project and sprint burndown we have used angular-chartjs which provides responsive charts with various other user interfaces display customization options

4. Database architecture

As we know the pure scrum is difficult to intrigarate with project management in real life we normally incorporate some changes to make it work in the real projects. To make it realistic we also have done some realistic approaches. Here we will explain those changes we have made. Which is also necessary to understand the functionality and how the application works.
We have an entity which we name as Project and other as Product. Currently we have one to one relation between a project and product but with our design project and product can have many to many relation. This is because in real life different team under different project name can work together to build a product if the product is huge. And as a product is finished building the project did not get closed. The project and it team member may start developing another product under same project for same client. Project contains the customer details and the team information and product contains the entities related to the product in scrum as stories sprint and tasks etc.
We have used Mongodb to develop our database. where entity type is saved under its own collection. In each collection each document which represents an unique entity has its own mongodb generated ID field which uni-

quely identifies the document. The uniquely identified entities are user, project, product, sprint, stories, tasks , definition of done.



1. Project collection contains project basic details and it manages the below collections:
   a) Definition of done collection: contains the definition of done which is applicable for each story to be check as implemented correctly. We associate the definition of done with the project so that if needed a definition of done can be included or excluded in the product which is build under the project. Project contains a defOfDones array which contain the definition of done Ids which are crated under the project.
   b) Team member details which is stored under the access request table. Access request table has columns which identifies which members are assigned to a particular project. Users get access to view or modify entities related to the project he has access in.
2. Product collection contains all the details corresponding to the product we build using scrum.
   a) We keep product basic details as shown in ER diagram. we associate the product which is build under the project using the field 'ProjectId' in product collection.
   b) We keep the story id in the 'Stories' sun collection, which represent one to many relation with product and stories. This sub collection contains the stories which are included for building the product.
3. We also keep the milestone details using one to few modeling technique as per mongodb where we keep all the details of milestones inside product collection a sub collection. this sub collection does not have unique identity and only can be identified with Product it is associated with.
4. The user table contains the user details like user name and user id user password and his privilege(explained in page x) in the application using 'role' field. an user is uniquely identified by its email id. this table is used to user login verification and new user creation. Accessrequest collection does not has a reorganization as an entity but this is used to build several functionalities for the application. The access requests raised by user to get access for any project is kept and managed using this project. It contains a user has access in which project(s), it also contains which role user has in the project. user

can have Developer, scrum master or owner role. This collection is also used to keep the information to check which project user had accessed at the time of last logout. This information is used to provide the default view for a project next time user logs in. To act upon a raised request by Project owner also this collection is used.

5. Stories: stories collection contains all the stories which are created under the project. The basic details as shown in diagram. All the stories are created under a particular project. Project act as super set of stories which are building under the product. stories created under a project can be included or excluded in/ from the product as needed which is being build in the project.
   The 'projectid' field associated the story with the project. The 'assignedtoProduct' field identifies if the story is included in the product.
   The 'status' field identifies the status of the story which can be in open if its not being developed under any sprint. 'In progress' if the story is being developed under any print. 'Closed' denotes the story is developed completely.
   The 'Progress' field is used to show the progress of the story.
6. Sprints: Sprint collection contains the details corresponding to the sprints. Other tan the basic details it contains the product id under which the sprint is created and a sub collection which contains the stories being developed under the sprint.
7. Task: tasks collections contains the basic details of tasks including the userid whom the task is assigned to and the task id under which the task is created.

5. User documentation

As mentioned earlier the ScrumHelper tool enables the efficient execution and management of projects. This section give the user guidelines on the usage of the product . The subsections provide the actions allowed by the users.

User Roles

- Admin
- Project Owner
- Developer
- Scrum Master

Admin - The admin is the administrator of the ScrumHelper tool. The admin user provide the project owner rights to any user who wish to be project owner. As per the process this request is send to the admin using mail. The role assignment is performed from the ScrumHelper tool by the admin user.

Project Owner - The project owner can create new projects in the ScrumHelper tool and add team members to the new the project who would be working on the project. The project owner also assigns roles to the team members in the project.

Developer - The developer role is assigned to the users who would be working on the project.

Scrum Master - The scrum master role is assigned for the user who would be adding impediments to the project.

Here we the structure and the functionality of the project are explained.

There is a ***Home page*** which displayed when the web application is requested. In the Home page title bar there are tabs to navigate to About, FAQ, Contact, Signup and Login page. *(Refer Manual page 2)*

About Page: contains the details about the website and the details of the developers. The announcements regarding upgrades of websites and cautions. *(Refer Manual page 3)*

FAQ Page: contains the details about why the web application should be used. *(Refer Manual page 4)*

Contacts Page: contains the details of the contact for the website development organization with the location details shown using google maps API . An option to send message to the web application development organization is also available. *(Refer Manual page 5)*

Signup Page: Sign up page enables user to signup for the first time into the website. Once a user is signed up an unique entry is made in the users table with default privilege as 'user'. This page has been built according to the requirement where a user can register to the system with user name, email and password. *(Refer Manual page 6)*

Login page: Using login page user can login with his email id and password. *(Refer Manual page 7)*

Inner Home page:
Once a user is logs in which view he/she gets depends on the privilege he has for the project. user can access the data for the project he/she has approved access. The last project with role he/she has set before login last login decides details of which project is shown as the user logs in.

All user can access the pages irrespective what privilege he/she has. The inner Home page, My projects, My Tasks, Definition of done, Access tickets, Change Password, Change View. These pages can be navigated using the tabs in the top title bar. *(Refer Manual page 9)*

Change password: After the user logs in successfully and wishes to change password. This option is available in the dropdown menu under user name in title bar *(Refer Manual page 8)*

In the inner home page always the product burndown chart and the open tasks of the project which user viewed last time before logout are shown. *(Refer Manual page 9)*

If user has scrum master privilege he/she is able to access the below pages using the left side navigation bar. *(*Refer Manual page 10)*

- Under Product tab Product Backlog and Product Burndown.
- Under Stories tab View stories.
- Under Sprint tab sprint backlog, sprit burndown view sprint histories.
- Under Tasks tab View Tasks.

If user has user privilege he/she is able to access the below pages using the left side navigation bar.
- Under Product tab Product Backlog and Product Burndown.
- Under Stories tab View stories.
- Under Sprint tab sprint backlog, sprit burndown view sprint histories, Create sprint and Manage Sprint.
- Under Tasks tab View Tasks. Create Task and manage Task.

If user has owner privilege he/she is able to access the below pages using the left side navigation bar.
- Under Product tab Product Backlog, Product Burndown, Product Create and Product Manage.
- Under Stories tab View Stories, Create Stories and Manage Stories.
- Under Sprint tab sprint backlog, sprit burndown and View Sprint Histories.
- Under Tasks tab View Tasks.
- Under Project tab Manage Project.

If user has admin privilege he/she is able to access the below pages using the left side navigation bar.
- Under Product tab Product Backlog, Product Burndown.
- Under Stories tab View Stories.
- Under Sprint tab sprint backlog, sprit burn-down and View Sprint Histories.
- Under Tasks tab View Tasks.
- In addition he/she gets an extra tab in top title bar named Admin.

There should be one admin whose task is basically to provide privileges to the users who has an account in the application. *(Refer Manual page 11)*

Home page also shown the ***current updates , online or offline users and the chat facility*** with the online users in the right side of the page. *(Refer Manual page 12)*

My Projects Page:
This page contains the details of the project for which an user has approved access. The below details of the project are shown.

Project Id, Name, team size number of Sprints for the project, Project start and end date.

The page also has a print button to print the details. *(Refer Manual page 13)*

My Tasks Page:
This page contains the details of the tasks which are assigned to the user. The below details of tasks are shown.

Project Name, Task Number, Task description, Start and end ate of task , Task Priority, Task Status, who has assigned the task and the story id under which the task is created.

The page also has a print button to print the details. *(Refer Manual page 14)*

Definition of done Page:
A separate page is designed to display the definition of done details due to its importance which are often forgotten. It contains the project name under which definition is created its Id and description. because definition of done applies to all the tasks that has been done according to its category this is relevant for all the project tasks and for all the member with different roles in the project. As everyone has to have access more often to this information. *(Refer Manual page 15)*

Access Tickets Page:
This page is used to raise and manage accesses to the project. *(Refer Manual page 16)*

- Raise request tab: User has to raise request with the project name and the role for which he/she wants access.
- View Request: tab: User can view the details of the raised request and its status.
- Manage Request: This page shows the details in tabular format of the requests in which the user has approved access. with who has raised the access for the project. But only Owner of the project can approve or reject the requests.

Admin Page:
This page provides the admin of the website to manage priviledges for the enrolled users.
This page shows the user name emain address, its current proviledge in tabular format. It also provides admin to change proviledge of each user and delete a user from the application database.
User with different proviledge can access different pages as explained before. *(Refer Manual page 11)*

Product Backlog page:
This page shows the product backlog of the product/project which is set using the change view modal or the product/project details user had accessed before last login.
It shown Story number, Priority , description, story points, status of the story, product under which the story is assigned, Its Progress and user name who has created the story which is thee product owner apparently. *(Refer Manual page 17)*

Product Burndown page:
Shows the product burndown chart with ideal and actual burndown as a line graph. *(Refer Manual page 18)*

Product Create page:
This page is accessable by user who has owner priviledge. Once a project is created with stories. A product which will be developed under the projet can be created using this page. *(Refer Manual page 19)*

Basic details tab:

- Basic details of the produt can be given in this page while creating a product , and the project under which the product is created can be selected. it includes product id, name, description and the customer name. there is a option to assoiate the product with a project. upon selecting the project it shows the project start and end date automatically which can be updated.

Team details and DefofDone tab:

- shows the corresponding team details and definition of done data which is created under the project.

Create Milestones tab:

- Milestones are created for the product using this tab. Milestones Aare basically the releases which are planned for the product in scrum. Generally after each sprint a new version are not released. this can be tracked using this functionality. This page also shows all the already created milestones.

Assign Stories tab:

- Stories created udder project can be signed to the product which have to be built under it.

Product Manage page:

This page is accessible by user who has owner privilege. Once a product is created it can be managed and updated through this page. *(Refer Manual page 20)*

Basic details tab:

- Basic details of the product can be updated, only the associated project is not updatable and is fixed once a product is created.
- Team details and DefofDone tab: shows the corresponding team details and definition of done data which is created under the corresponding project. these details can only be updated from project manage page.

Create Milestones tab:

- Milestones can be created for the product using this tab. created milestones can also be deleted or edited. This page also shows all the already available milestones. Milestones status can be updated from this tab.

Assign Stories tab:

- This page shows all the stories under the project which are Stories created udder project which are assigned or can be assigned to the product. It contains story number, description field. The Assigned field shows whether the story is assigned to the project, assigned and closed or cab be assigned. The Manage column has view and include or exclude buttons. on clicking the view button the story details is shown below the table. If a story is not assigned to the project it can be assigned using the include(PLUS) button. If a tory is not already in closed status or already included in any sprint it can also be excluded using exclude (MINUS) button. A closed story does not provide option to include or exclude.

View Stories:

Details of all the stories assigned under the product can be view from this page. It containsStoy Id, Priority Description, story points, status, project under which the story is created , its progress is percentage, its start and end date and owner name which created the story. The progress of the story is calculated when a task under it is closed. it is the percentage of the story points completed by the closed tasks with total story points for the whole task. *(Refer Manual page 21)*

Create Stories:

This page shows the stories which are created under the project/product. It has a create button (PLUS) which is a modal to create a new story. It also has a print button to print the table shown in the page. *(Refer Manual page 22)*

Manage Stories:

Created stories can be updated from this page. if a story is in progress status which implies the stores is being worked on under s sprint its status only can be updated but not the other details.

A story which is In progress or closed cannot be removed using the remove link associated with every story in the table. *(Refer Manual page 23)*

Sprint Backlog page:

Sprint backlog page provide the view of any sprint under the current product view. There is option given to user to select a sprint and view the details of a print. it is very useful for current sprint. Upon selecting a sprint, it lists the stories which are developed under the sprint and the corresponding tasks. Tasks are categorized by its status and are displayed in one of the three columns. Open Tasks those are not yet started comes under ToDo column. Tasks in progress comes under InProgress column. And closed tasks come under done column. Each task dis-

played with its description and the corresponding task wheitage. A task status can be updated by clicking on the task, which open a modal and provides options for changing its status. *(Refer Manual page 24)*

Sprint Burndown page:
The burndown chart of each sprint is shown upon selection of a specific sprint. The sprint burndown is plotted with number of days in sprint in X axis versus effort in Y axis. Effort is calculated story points of tasks with Hour multiplier which is constant value decided by the project. *(Refer Manual page 25)*

View History page:
This page shows the details of all the old sprints in a tabular format. It contains the sprint number, name, start and end date, sprint duration, status and progress. It also shows the sprint backlog in a list format with list of stories implemented under the sprint. upon clicking on the story, its details are displayed in a popup modal. *(Refer Manual page 26)*

Create Sprint page:
A sprit can be created in the page. *(Refer Manual page 27)*

Basic Details tab:

- The first tab Basic Details contains the basic information's like Sprint number, name start and end date that can be entered. The product for which sprint is created in the current product which is in view which is set using the Change View popup window.

Sprint Backlog tab:

- This tab contains two tables, one represent product backlog, It displays the stories which are included in the product still open and can be included in a sprint. The Stories which are InProgress (included in any other sprint) or already closed is not shows here. The other table shows the backlog of current sprint. A story can be taken from product back log table and included into sprint backlog table. This can be done by dragging and dropping a story from one table to another. Removing a story from the spent and adding back to product log is also possible if the story is not already in closed status.

Sprint Manage page:
Sprint manages page shows all the sprints details in same tabular format like sprint History. But a backlog of a sprint can be updated using the page. On clicking on the update button corresponding to the print it shows the sprint and product backlog in a popup modal. where stories can be transferred between both backlogs. The constraints exampled in previous page also apply here. *(Refer Manual page 28)*
Sprint status can be updated using the status bottom which shows the current status of the corresponding sprint. A sprint which does not have any story assigned cannot be closed or cancelled. A sprint can be closed if all tasks included in it are closed.
There is a sprint remove option available for all sprints unless sprint is Closed.
Sprint progress is also shown here which is re calculated and updated whenever a story is closed.

Task View page:
Task View page shows the details of all the tasks for particular story. There is a dropdown selector available to select the story number of stories which are under current product view. *(Refer Manual page 29)*
Upon selecting the story number the story description and story point is shown. the tasks details is shown is a table. it contains task number, description , start and end date, priority, status (which can be open, In progress or closed), story points weight age for the task, whom the task is assigned and who has created the task.

Task Create page:
A task can be created here. The story under which task have to be created is chosen using a drop down list. Story which is closed cannot be selected for task creation. In this case a message is shown in the top right corner of the page. If a story is included in a sprint task can only be created under that story. In this case also a message is shown to user in the top right corner of the page. The Plus bottom gets activated for a valid story and upon clicking on it a pop up modal is opened and task details can be filled. It contains task number, description, start and end date, priority, status (which can be open, In progress or closed), story points weight age for the task. A drop down shows all the approved users who has developer role under the project in which the product is being

built. Using this drop down task can be assigned to a developer. It also shows the user as the assigner, this is populated automatically with the current user name.
The table present in the page shows all the tasks which are currently created using the task create button. *(Refer Manual page 30)*

Task Manage page:
All the created task can be managed here. The story is chosen using a drop down list under which user intend to manage the tasks. Upon selecting the story number the story description and story point is shown. the tasks details is shown is a table. The table shows details all the created tasks. A task can be removed if it's in open status. task details also can be updated using update link for each task in the table. *(Refer Manual page 31)*

Manage project page:
The link to navigate to the page is shown under Projects heading in the left side menu.
This option is available only for the users with owner privilege. Once a project is created the user becomes the owner of the project and he/she can update details of the project. or If a owner access is given to another user upon raising access request he/she can access this page. *(Refer Manual page 32)*

Basic Details tab:

- The basic details of a project can be enter in this tab. Upon selecting the Plus bottom a modal is opened which contains the fields Project Id, name team size, Number of sprints start and end date of the project. The number of Sprint restricts the total number of sprint which can be created to developed the product under this project. All the projects created by the user or the project for which he has owner access is listed with basic details in this page. the basic details of the project can be updated and a project can be removed from this page.

Member/role tab:

- Member can be added to the project by the owner using this page. Once a project is selected by the owner. It lists the entire approved project member with his role in the project in a table form. Each member can be removed from the project. Doing this action user will not have any access to the project and its corresponding product and product related details.

- A user can be added to the selected project using add new member button. Once the bottom is clicked it opens a modal which lists the users who does not already have been approved in this project. Member's role also can be assigned here. If owner assign a member using this modal a user automatically gets access to the project he/she does not have to raise a separate request using Access request page to get access to the project.

Definition Of Done tab:

- Upon selecting a project in which user has proper owner access. User can create definition of done with its description and number. It then shows all the existing definition of done or newly created definition of done in the project. They can be included in the product accordingly. A definition of done also can be removed.

5. User Guidelines

Refer Manual.