# Small group project self-assessment

**Team name**: *Dingo*

The table below contains a list of epics mentioned in the assignment. For each epic that the team produced features for, add a brief explanation outlining how the user can access and use it. You have been provided with some code that incorporates some of the features you need (though you can extend it). This self-assessment will be used to help the marker figure out how to access your system, so try to keep it clear and concise.

Do review the marking scheme for the small group project to understand how the project is assessed. You are reminded that the assignment is **not** to try and implement every single epic.

|  |  |
| --- | --- |
| **Epic** | **Implementation** |
| Epic 1  The web application should have user authentication and authorisation mechanisms. Users should be able to create accounts, log in, and manage their profiles. | *The features in this cell were provided in the scaffolding project.*   * *A home screen gives users the option to sign-up or login (URL/)* * *Users without account, who are not logged in, can sign up as new users (URL/sign\_up)* * *Users with an account can log in (URL/log\_in)* * *Logged in users can log out (URL/log\_out)* * *Logged in users can change their password (URL/password)* * *Logged in users can change their profile (URL/profile)* * *After completing an account related task, logged in users are redirected to a currently blank dashboard (URL/dashboard)* * *If a not-logged in user tries to access pages requiring login, they are redirected to the login page, and then to the page they were after following successful login.* * *A menu is available on all pages while users are logged in.* |
| No further implementation |
| Epic 2  Users can form teams, and invite team members to their team. | * On the main page of the website, there is a ‘Create’ button. Clicking it and ‘’New Team’ will open up a form to create a team. * A logged in user can input a non-null unique team name and select a non-empty set of team members then press submit to create a team. * Within the navbar there is a ‘Teams’ button to display a list of a users teams. * A logged in user can select a team name to display a view for that specific team and invite other users by selecting one from a drop down menu and pressing the ‘Invite’ button. |
| Epic 3  Each user can create tasks, assign tasks to other team members, and set due dates. | * On the main page of the website, there is a ‘Create’ button. Clicking ‘New Task’, which is underneath the ‘Create’ button will open a form. * There are 2 pages for the form. * The first page allows the user to enter a name, description, deadline, team, and priority. * The user must select from one of the teams they are part of. * The second page of the form allows the user to select the members from their team, that they want to assign to the task. * The ‘Previous’ button moves the form to the previous page and ‘Submit’ moves it to the next page and sends the form. |
| Epic 4  Users provide a dashboard to view and manage assigned tasks, and team tasks. | * On the main page of the application (which the user is redirected to after logging-in and can be reached by clicking the ‘Task Manager’ heading in the navbar) the user can see a condensed list of tasks. * The 10 tasks with the closest deadlines are shown on the left and any overdue tasks are shown on the right. * The user can see a full list of all of their tasks by clicking ‘Tasks’ in the navbar. * The user can see a full list of all the teams they are part of by clicking ‘Teams’ in the navbar. |
| Epic 5  Enable users to search, order and filter tasks. These could be based on name, completion status, priority, due date ranges, assigned developer, or team/project. | * On the task list page, it is possible to sort tasks using the form just below the main heading. * User can select what to sort by, whether to sort in ascending or descending order, what to filter for and can search for tasks using the search box. * The filter search box is optional. |
| Epic 6  Add a priority system for tasks, allowing users to assign priority levels and filter tasks based on priority. Include automated reminders for high-priority tasks nearing their due dates. | * On creation, tasks can be assigned one of 5 priority levels (backlog, low, medium, high or urgent). * Can also modify the priority of an existing task through the ‘modify task’ form on the task detail page (reached by clicking on the task in the task list). |
| Epic 7  Introduce task dependencies, enabling users to link tasks and establish dependencies between them. Ensure that the application adjusts task timelines based on dependencies. | Not implemented |
| Epic 8  Integrate a time tracking feature for tasks, allowing users to log time spent on each task. Provide summary reports of time spent on tasks over different periods. | * On task detail page, User can log start and end time of each task. It also records the duration automatically. And a user can log many times. * Users in a same task can view each other’s logged time. * A validation is implemented to make sure start time users choose is before end time. |
| Epic 9  Implement an activity log that records user actions within the application, providing an audit trail for accountability. | Not implemented |
| Epic 10  Add a notification system (within the constraints of a Django) to alert users about task assignments and upcoming due dates. | Not implemented |
| Epic 11  Introduce gamification elements to motivate users, such as achievement badges, leaderboards, or a points system based on task completion. | Not implemented |