# Small group project self-assessment

**Team name**: *Rhinoceros*

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. | * User can view all their teams with the ‘My Teams’ tab in the dashboard, which includes both those created by them or those they have been added to. (URL/dashboard/my\_teams) * User can create a new team, saving the new teams details. (URL/dashboard/create\_team) * Users can add users from the database to their team, using the edit team button. (URL/dashboard/my\_teams/edit\_team/<int:team\_id>), where the team\_id corresponds to a specific team. * Users can view their added team members from the team info page, using the info button. (URL/dashboard/my\_teams/team\_info/<int:team\_id>), where the team\_id corresponds to a specific team. |
| Epic 3  Each user can create tasks, assign tasks to other team members, and set due dates. | * Users can view their personal tasks with the ‘My Tasks’ tab. (URL/dashboard/my\_tasks) * Users can view their team tasks within a specific team – in the view accessed by the team info button. (URL/dashboard/my\_teams/team\_info/<int:team\_id>) * Users can create a task for a specific team in the team info view. (URL/dashboard/create\_task/<int:team\_id>), where the team\_id corresponds to a specific team. * Users can assign already created tasks to a team member with the edit task button. (URL/dashboard/ my\_tasks/edit\_task/<int:task\_id>), where the task\_id corresponds to a specific existing task. * Users can set due dates when creating a new task, and can edit the due date of an already created task. |
| Epic 4  Users provide a dashboard to view and manage assigned tasks, and team tasks. | * Users can view and access all tasks (personal and team specific) and teams from the dashboard, navigating with the relevant tabs and buttons. |
| 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. | Not implemented |
| 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. | Not implemented |
| 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. | Not implemented |
| 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 |