**Tasks List Management - Project**

1. **Static content (View)**

* Html files
* Stylesheet files
* JavaScript files

1. Front page: should include Links for login pages for
   1. Admin
   2. Project manager,
   3. Users
2. Login pages for each user
3. Login error page => a single page for all users.
4. Pages for each user => includes all options for each user, including logout
5. Controller – All servlet files
6. Model – all other java objects, such as users
7. Database –

**Use cases present in Cameron code:**

|  |  |
| --- | --- |
| Users create new tasks, including due dates and task categories | √ |
| Users edit tasks | √ |
| Users delete tasks | √ |
| Users view their task list | √ |
| Overdue tasks are highlighted to users | √ |
| Users can mark tasks as complete | √ |
|  |  |

Potential classes

1. User

* Name
* Id
* email
* address/location
* task list
* team id

1. Task

**Servlet Filter**

1. Order set in web.xml file.
2. Can be used to check if a session is available, e.g, if user is login. If not, create a session – forward/redirect to login page.
3. Authentication – session validate/invalidate.
4. Authorization – get user roles and provide access to the appropriate page(url).
5. Logout – session invalidate in servlet.
6. Page security
7. HTTPSession listener - to get notification when a user logs in or logs out, or any other event activity.
8. **Promise** for asynchronous callbacks – it has highest priority vs what? **setTimeout**
9. Compared to ajax call, fetch() api returns Promise.
10. Use application context to count the number of visits to a site.