Requirements for a proposed Project Management System

Charles Pascoe

April 6, 2017

Contents

1 Inception

1.1 Problem Definition

Today's competitive markets have lead to the growth of highly distributed teams; it is no longer enough to simply employ people from a small catchment area around a company's premises. Instead, companies will find and employ the best candidates from all over the country, or even all around the world, and use modern communication technologies like BT's One Voice service to bring everyone together at an affordable cost.

While communication is a key element of teamworking, it is equally important to keep track of the work that needs to be done, who is responsible for delivering it, and what has been completed so far. A solution is needed that will help distributed teams track and manage multiple projects and tasks.

In addition to being geographically dispersed, many people today work 'on-the-go' such as when commuting on the train; in these environments, network connections can be slow, or they might not have any internet access at all. Thus, any solution that aims to assist the modern workforce should use minimal network bandwidth, and ideally work offline.

Finally, some companies use 'bring your own device' policies, which reduce the cost of maintaining devices for the company, but restricts the selection of software solutions the company can choose from, as they have no control over the operating system of the device. This usually means that web-based solutions are preferred.

1.2 Prior Art

Trello is a simple and flexible general-purpose project management tool, and the basic package is free. It uses the concept of virtual cards on boards, with dynamic content and the ability to embed documents and media. However, since it is a general-purpose solution, there is a lack of awareness of business processes, meaning that the user has to enforce the processes manually, which is error-prone and cumbersome to use in anything other than small teams or individuals. Also, it has very basic authorisation, meaning that anyone with access to the project can modify anything.

Another popular project management system is Jira, by Atlassian. It boasts numerous plugins that extend its functionality, and comprehensive REST APIs that enable other systems to interact with it, allowing more automation in business processes. However, the user interface does not work well on mobile devices, and is highly dependent on internet connection to function. Clearly, a new solution is needed in order to solve the problem definition.

1.3 Solution Scope

The proposed solution is a project management system in the form of a single-page web application. It will work in all modern browsers across a number of different platforms, and if designed and built correctly, should have minimal network requirements after the application has loaded - there is even the possibility for the application to run offline. This would solve the problems involving limited network connectivity and targeting multiple device platforms. Distributed teams also have a strong requirement for communication; however, this solution will not focus on communication aspect, but rather on the tracking of work that needs to be done.

It is worth noting that many organisations have their own security and performance requirements of the software they use; therefore, the solution will be provided as a server 'package', which they can configure and deploy as they please. This will allow them to use a hosting service of their choice - either an approved external service or their organisation's internal hosting - and to scale it to meet the number of users of the system. Additionally, by allowing teams to host it themselves, it reduces the impact of compromising the system; if the solution was instead provided as a hosted web service, the reward for compromising the system is much greater, since the system would contain a significant amount of data of many different organisations, and so an attacker would be willing to use more resources to break into the system. By hosting it themselves, there is less of a reward for compromising the system.

1.4 Stakeholders

The proposed solution will give project leaders/administrators a high-level view of the work in the project, a view of how the work is distributed throughout the team, and it will help them to plan work in the long term. The distribution of work is especially important in geographically distributed teams, as they won't be able to have the same level of communication that a team who work together in an office would have. This could lead to the scenario where some team members have too much work whilst others don't have enough to do.

Team members will also benefit, as the system will give them a clear definition of what work needs to be done, and who should be doing it. As a web application, it will work on all modern web browsers across numerous platforms, and so will work on a number of devices that they use for work. The low network bandwidth requirements will allow them to use the mobile data network, allowing them to work wherever they want to.

Additionally, higher management or end customers could benefit from the system, as they could be allowed read-only access to the system to view statistics of the work done on their project in order to follow its progress.

2 Requirements

The requirements are defined in a number of forms: user stories (US), business logic requirements (BLR), security requirements (SR), and information requirements (IR). All requirements are given numeric identifiers (e.g. SR-2.1) to simplify referencing the requirements. These requirements cover what is explicitly required by the end user, and does not cover implicit requirements such as legal obligations (e.g. as accessibility and the Data Protection Act); the system will also comply to these, as will be covered during the design and build phase of the development.

2.1 User Stories

Requirements should be driven by the needs of the user; therefore, most of the functional requirements will be covered by user stories. User stories are short descriptions of what a particular user want to achieve using the system. They are typically used in agile development as the catalyst for discussion between the customer and the development team; however, they are still suitable in this case where there is a potential market rather than a particular customer, as they provide and excellent means of modelling what a user would need from the system in simple and understandable language.

In addition to the user stories, other details and business logic requirements are recorded as well; these define functional constraints or required functionality essential to the business problem that the application solves.

2.1.1 System User Stories

The system user actor represents all users of the system; although actors are distinct, this actor has been created to prevent repetition of common user stories, such as login and changing the user password.

US-1.1: As a user, I want to set my password when I first use the system

- **BLR-1:** Set/Reset password must be authenticated via email
- Additionally, SR-2.1 defines password complexity requirements

US-1.2: As a user, I want to log into the system

US-1.3: As a user, I want to change my password

2.1.2 System Administrator Stories

System administrators are responsible for managing the system itself, including who has access to the system and which projects the users have access to. They will also be responsible for maintaining the server the system is running on, although not through the system itself. Additionally, they will need to review output information to ensure that the system is running as expected and that there is no unusual activity.

Since teams will be hosting it themselves, it will also likely be the case the system administrators are regular users of the system (workers, project administrators, etc.); however, for the sake of the user stories, system administrators are only responsible for high-level management of the system.

US-2.1: As a system administrator, I want to add users to the system

US-2.2: As a system administrator, I want to deactivate users of the system

US-2.3: As a system administrator, I want to define the permissions of each role

- SR-6.1 requires that project members have a role that controls what each member is permitted to do within the project

US-2.4: As a system administrator, I want to create a project

US-2.5: As a system administrator, I want to add a user to the list of project members

- **BLR-2:** A user can only exist once in the list of project members

US-2.6: As a system administrator, I want to change the role a member has within the project

US-2.7: As a system administrator, I want to remove a member from the project

US-2.8: As a system administrator, I want to view and filter log output information of the system

- This is a lower priority, as the system administrators will still be able to access this information without it being exposed directly through the user interface

2.1.3 Project Administrator Stories

Project administrators are responsible for managing the members of the project, the roles of the members, and the distribution of work among those members. It may be the case in reality that project administrators also work on tasks (as it is common to have 'worker-managers'); however, for the sake of the user stories, project administrators are only responsible for high-level management of the project.

US-3.1: As a project administrator, I want to add a user to the project members

US-3.2: As a project administrator, I want to change the role a member has within the project

US-3.3: As a project administrator, I want to remove a member from the project

US-3.4: As a project administrator, I want to see a summary of work in a project

- The summary must show to the project administrator the quantity of work, and the estimated time that is needed to complete it. IR-4 defines the information that is stored with each task. Exactly what information is shown and how it is presented will be decided in the user interface designs.

US-3.5: As a project administrator, I want to see all incomplete tasks within the project

US-3.6: As a project administrator, I want to see all completed tasks within the project

US-3.7: As a project administrator, I want to view a task within the project

US-3.8: As a project administrator, I want to change who is assigned to a task

- **BLR-3:** Only members of the task's project can be assigned to the task
- This includes removing the assignee, which puts the task into an 'Unassigned' state

US-3.9: As a project administrator, I want to Change the state of the task

- **BLR-4:** Only the assigned worker or project administrators can change the state
- As described in IR-5, the task state is one of 'Open', 'In Progress', or 'Complete'

2.1.4 Worker Stories

Workers are the people primarily doing the work in a project. They will want to access the system so that they can see the work they need to do, to track the effort invested in completing a task, to create new tasks, log effort against tasks, and marking tasks as complete. The task management aspect needs to have little overhead so that it doesn't interfere with the work itself.

US-4.1: As a worker, I want to see all projects that I can access

- This should show a summary of the tasks assigned to them in each project (e.g. number of tasks assigned to them, and the total amount of effort remaining)
- Also, SR-5.7 requires that users cannot see projects (or the tasks within those projects) that they are not members of

US-4.2: As a worker, I want to see all tasks within a project that I can access

US-4.3: As a worker, I want to see all tasks assigned to me in a project

US-4.4: As a worker, I want to see unassigned tasks in a project

US-4.5: As a worker, I want to assign an unassigned task to myself

US-4.6: As a worker, I want to view a task within a project

US-4.7: As a worker, I want to transfer a task to another worker by assigning it to them

- This includes removing themselves as the assignee

US-4.8: As a worker, I want to edit a task

- **BLR-5:** A worker can only edit:
  - Summary
  - Description
  - Priority
  - Effort Estimate
  - Target Completion Date

US-4.9: As a worker, I want to log effort against a task

- **BLR-6:** Only the worker assigned to the task can log effort

US-4.10: As a worker, I want to change the state of the task

2.1.5 Observer Stories

Observers want to access the system to view aggregate data/statistics, and they are not necessarily interested in low-level details. They won't be entering data into the system, but rather viewing data presented to them.

US-5.1 As an observer, I want to See all projects that I can access

US-5.2 As an observer, I want to See all tasks within a project that I can access

US-5.3 As an observer, I want to View a task within a project

- **BLR-7:** An observer cannot see the tasks of a project they are not assigned to

US-5.4 As an observer, I want to See a summary of the work that has been completed and the work that still needs to be done

- Exactly what will be included in this summary will be decided when creating the user interface designs

2.2 Security Requirements

The security requirements have been created using threat analysis, which involves identifying threat actors to model potential adversaries - focussing on their motives and capabilities - and then identifying each threat they pose, analysing its probability and impact, and proposing countermeasures in the form of security requirements. This approach has been chosen because it ensures that the security requirements are relevant and complete, and it also documents the rationale behind the security requirements.

Risk is considered a product of the probability and impact; a threat with high probability and high impact would be high risk, and thus its countermeasures are more important - this inverse is also true, in that low probability and low impact have low risk. Still, there could be a threat which has high probability and low impact, or low probability and high impact; such threats would have a medium risk, and it requires careful thought as to what countermeasures need be implemented, since it requires effort (and thus money) to implement them. By taking risk into consideration, the security requirements have been ranked by how important its implementation is using the keywords MUST, SHOULD, and COULD.

2.2.1 Threat Actors

A *passive eavesdropper* can intercept traffic, but does not modify it. Their aim is to collect information of value (e.g. for corporate espionage), or sufficient information to gain access to the system, thus becoming an active attacker.

An *active attacker* is a (typically) unauthenticated, unauthorised actor who actively attempts to break into a system, or cause it to function improperly, by making specially crafted requests to the server. Their motivation can range from accessing valuable information from the system, preventing legitimate users from accessing the system out of spite (either to the organisation or the team in order to hinder their work), or using the server as a platform for further attacks on other systems.

A *malicious user* is an authenticated user of the system who may abuse their access to the system. They could be motivated by wanting to sabotage a project for any number of reasons (e.g. about to lose job, severe altercation with project lead, etc.), or bribed by another organisation for corporate espionage. This is the hardest threat actor to defend against, as they need to have access to the system in order to do their job.

2.2.2 Threat Analysis and Countermeasures

Threat 1: Interception of traffic by a passive eavesdropper

Probability: *High*, because the users could be anywhere in the world, they will be connecting over the internet (which can hardly be considered secure in itself) over a number of different connection methods such as public WiFi hotspots, commonly found in Cafés or on Train and Bus services

Impact: *High*, because it allows unauthenticated and unauthorised access of data, and if an eavesdropper intercepts authentication information, they could do anything that user can do - in the case of a system administrator, total control of the system

Countermeasures:

- **SR-1.1:** It MUST use HTTPS, configured with suitable key-exchanges and ciphers

Threat 2: Network password cracking by an active attacker

Probability: *High*, as there are a number of password-cracking bots which will try to login to many different sites using common password dictionaries, such as the ones provided by Yahoo

Impact: *High*, as it would allow a malicious attacker to login as an authorised user, capable of doing anything that user can do, which in the case of a system administrator is control the whole system

Countermeasures:

- **SR-2.1:** All passwords MUST meet the following requirements:
  - Be between 12 and 256 characters
  - Have at least one uppercase character
  - Have at least one lowercase character
  - Have at least one number
  - Have at least one non-alphanumeric character
- **SR-2.2:** The system MUST not allow weak passwords, by checking the password against a password blacklist of common passwords
- **SR-2.3:** The system SHOULD block repeated login requests
- **SR-2.4:** Passwords COULD expire, and force the user to set a new one after a set period of time
- **SR-2.5:** It COULD use two-factor authentication, which would prevent login even with the correct password

Threat 3: Unauthenticated active attacker gleaning information by timing server responses

It is possible to gain information about a system based upon average response times; for example, if an active attacker repeatedly tried login requests and timing the response, they would find that non-existent users would result in a relatively short response time, as it only requires a simple query to the database. However, if a user is present, the server will have to hash the provided password, probably using a slow hashing algorithm such as BCrypt; although the given password will almost certainly be wrong and thus return the appropriate response code, the attacker will know that the user is present in the database since the response takes longer, and thus they can try executing a dictionary password attack. This is one possible way an unauthenticated attacker could gain information about the system using response timing.

Probability: *Medium*, as it requires effort to either programmatically or manually interpret information such as response times

Impact: *Low*, as it wouldn't reveal stored data, but could indicate the internal operation of the system or the presence of certain data

Countermeasures:

- **SR-3.1:** The system COULD conceal the actual response times on unauthenticated and unauthorised requests

Threat 4: The database could be compromised by an active attacker, or data could be leaked by the system due to an active attacker manipulating the system (e.g. SQL injection on login page)

Probability: *High*, as there are automated bots that will check for exposed database ports or attempt SQL injection (or similar techniques), and there are many SQL injection scripts/tools readily available for even inexperienced 'hackers'

Impact: *High*, as it could result in unauthorised access, modification, or loss of data

Countermeasures:

- **SR-4.1:** The system MUST prevent SQL injection if using an SQL database
- **SR-4.2:** The system MUST access the database using unprivileged authentication credentials, such that it is not able to cause serious loss of data
- **SR-4.3:** Authentication information MUST be hashed using suitable hashing algorithms
- **SR-4.4:** The system MUST use suitably slow hashing algorithms for low-entropy authentication information such as passwords

Threat 5: The user could log into multiple devices, and one device gets compromised either virtually (i.e. the device gets remotely hacked) or physically (e.g. the device gets stolen), or the user logs into a public device (e.g. library/coffee shop)

Probability: *Medium*, because the users could be using a mobile device anywhere, and it is possible for someone to steal the device or access it when the device is left unattended

Impact: *High*, as it would allow a malicious attacker to use the system as an authorised user, capable of doing anything that user can do, which in the case of a system administrator is control the whole system

Countermeasures:

- **SR-5.1:** Users MUST have multiple login sessions with unique authentication information
- **SR-5.2:** Login sessions MUST expire after an appropriate period of inactivity, and the user must be able to control how long the session lasts (e.g. 'Remember me' checkbox on login)
- **SR-5.3:** System administrator users MUST request elevation, providing their username and password, to do privileged actions
- **SR-5.4:** System administrator elevation MUST expire after a certain time period or be dropped manually by the user
- **SR-5.5:** System administrator elevation MUST only apply to that login session
- **SR-5.6:** All login sessions for a user MUST expire when the user's password is changed (except the login session that the user is currently using)
- **SR-5.7:** Users MUST not be able to see the projects that they are not members of
  - This minimises the impact of an account being compromised to only the projects that user is assigned to
- **SR-5.8:** The system MUST log sufficient information to allow auditing (for monitoring unusual user activity etc.)
- **SR-5.9:** The system SHOULD allow individual logins to be revoked by the user or by the system administrator

Threat 6: A user may accidentally (or deliberately, in the case of a malicious user) make serious changes to a project

Probability: *High*, because it is very likely that there will be new users to the system who are unaware of implications of their actions; also, malicious users or compromised user accounts could cause significant damage

Impact: *High*, because it would result in the unauthorised access, modification, or loss of data

Countermeasures:

- **SR-6.1:** All users MUST have a role in a project that controls what permissions that user has (i.e. what they can do)
- **SR-6.2:** Roles and their permissions SHOULD be configurable by the system administrator, according to the principle of least privilege

2.2.3 Additional Security Requirements and Constraints

Given that the system must use an expensive hashing algorithm for low-entropy authentication information (SR-4.4), the password cannot be used to authenticate individual requests to the server, as there is too much computational overhead. Therefore, an OAuth2-style token system (or similar) should be used, as long as its implementation still meets the requirement that all authentication information should be hashed (SR-4.3).

2.3 Information Requirements

The information requirements cover what the system needs to contain, and the relationships between the different types of data. Although a functioning database has been designed and created, these revised information requirements - combined with the user stories, business logic requirements, and security requirements - will result in changes to the database. These changes will be documented in the design and build phase.

IR-1: It must store information about the system users, including:

- User's name
- Email
- Password Hash
- Whether or not the user is a system administrator
- Set/Reset password token

IR-2: It must be able to store the hashed authentication information for each user, in such a way so as to allow multiple independent sessions per user

IR-3: It must store information about the projects, including:

- Project key (a short sequence of characters used to identify which project a task belongs to)
- Project name (a brief, meaningful description of the project)
- Whether or not the project is complete
- When the project was completed

IR-4: It must store information about the tasks, including:

- Summary
- Description
- Target completion date
- Actual completion date
- Estimated effort to complete

- Priority
- A log of work against that task

IR-5: Each task exists in one state, either ‘Open’, ‘In Progress’, or ‘Completed’

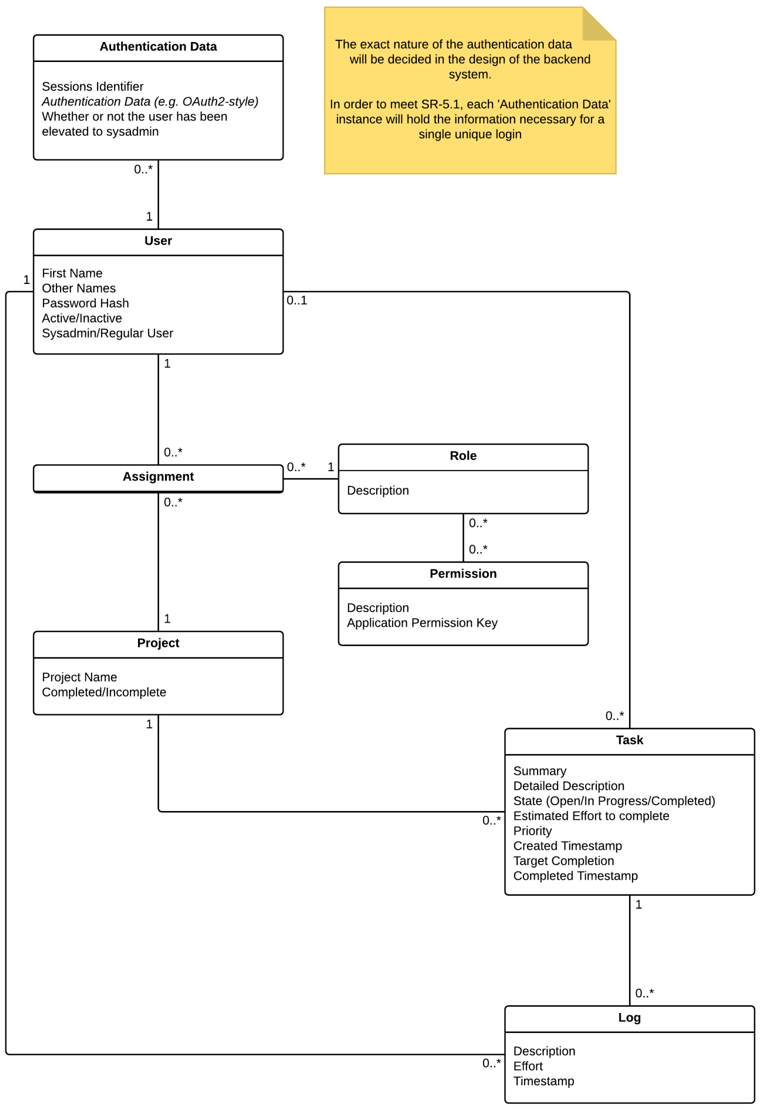
IR-6: Each project may have many people assigned to it, and each user may be assigned to many projects

IR-7: Each user has exactly one role in a project, which controls what that user can do within the project

IR-8: Each role has a set of application-defined permissions which define what a user in that role can do within the project

IR-9: Each project may have many tasks, and each task must belong to one project

IR-10: Each task can either be unassigned, or assigned to one user



Authentication Data
Sessions Identifier
Authentication Data (e.g. OAuth2-style)
Whether or not the user has been elevated to sysadmin
The exact nature of the authentication data will be decided in the design of the backend system.
In order to meet SR-5.1, each 'Authentication Data' instance will hold the information necessary for a single unique login
0..*
1
User
First Name
Other Names
Password Hash
Active/Inactive
Sysadmin/Regular User
1
0..1
1
0..*
Assignment
0..*
1
Role
Description
0..*
0..*
0..*
Permission
Description
Application Permission Key
1
Project
Project Name
Completed/Incomplete
1
0..*
Task
Summary
Detailed Description
State (Open/In Progress/Completed)
Estimated Effort to complete
Priority
Created Timestamp
Target Completion
Completed Timestamp
0..*
1
0..*
Log
Description
Effort
Timestamp
0..*

A simple UML representation of the information the system needs to contain

Appendix

Glossary

- **API:** An Application Programming Interface is a well-defined set of methods or procedures that allow access to or modification of data within a system, and conceals the exact implementation of how those actions are achieved
- **Effort:** "Work" and "Effort" are used interchangeably to mean the amount of time needed to complete a task, or the amount of time already spent on the task. Time is used to quantify the effort put into completing the task.
- **HTTPS:** Hypertext Transfer Protocol over Transport Layer Security (TLS, successor to Secure Sockets Layer, SSL), which uses cryptography to ensure confidentiality, integrity, and authenticity (usually of the server, in the form of a certificate)
- **Project Member:** A user that is assigned to a project, each project member has exactly one role in the system
- **REST:** Representational State Transfer is an architecture for client-server communications. Essentially, it focusses on entities (identified by Universal Resource Identifiers, or URIs) and operations on those entities, and ideally should not require a special sequence of communications to achieve one thing (i.e. there is no state on the server, all valid and authorised requests should be available)
- **Role:** Defines a set of permissions a project member has within a project
- **Slow Hashing Algorithm:** A one-way algorithm for generating a message digest (or 'hash') of input information, which deliberately uses extra computation to increase the resources needed to compute a single digest. Typically used with low-entropy (i.e. predictable) data such as passwords.
- **Work:** See "Effort"