***Requirements Document for E-Magnet :***

*Kshitij Kaushik*

*Pooja Gaikwad*

*Sandeep Bethi*

*Priyabrata Purohit*

*Umeaz Kheradia*

*Apoorva Joshi*

Table of Contents

1. Introduction 2

1.1 Purpose of this document 2

1.2 Reference Materials 5

1.3 Specific Terms and Acronyms 5

1.4 User Roles 5

1.5 Assumptions 8

1.6 Constraints 9

1.7 Dependencies 9

2. Requirements 10

2.1. Business Requirements 10

2.2. User and Functional Requirements 10

2.3 Nonfunctional Requirements 11

2.4 Reporting Requirements:- 13

2.5 System and Integration Requirements:- 13

2.6 Security Requirements:- 13

Revision History 14

# Introduction:

This document contains the system level and user level requirements for the E-Magnet system. The E-Magnet is a software application used for automating manual filling of time sheets. E-Magnet tool provides a common platform for all the members a project team. This platform provides the user with features like creating a task with existing management tools (in our case JIRA) used by the company, getting notifications, scheduling appointments with fellow team members and getting commit notifications and alerts. All these requirements have been derived from several different sources.

## 1.1 Purpose of this document:

This document is intended to guide development of the E-magnet tool. This document contains all the system level and user level requirements for this project. It will go through several stages during the course of the project:

1. **Draft**: The draft version, is compiled after requirements have been discovered, recorded, classified, and prioritized.

2. **Proposed:** The draft document is then proposed as a potential requirements specification for the project. The proposed document is reviewed by several parties, who may comment on any requirements, either to agree, to disagree, or to identify missing requirements. Readers include end-users, developers, project managers, and any other stakeholders. The document may be revised and re-proposed several times before moving to the next stage.

3. **Validated:** The document is validated, once the various stakeholders have agreed to the requirements in the document.

4. **Approved:** All the stakeholders accept the validated document, as an appropriate statement of requirements for the project. The developers then use the requirements document as a guide for implementing the project. It is also used to check the progress of the project as it develops. This document must establish an agreement between the customer and suppliers on what the system must do. It should contain enough detail to provide a basis for estimating cost and budgets. Each version should provide a baseline for verification and validation. It should facilitate transfer of knowledge, commitment, and intent and finally the usability of tool to the stakeholders. The final version of this document should accurately and completely describe the final product and serve as a basis for future enhancements. Both the writer and the reader should strive for these goals in the elicitation, analysis, presentation and validation of this document. Through base lining, versions of this document should clearly identify all changes.

**How to Use This Document:**

We expect that people with different skill sets will use this document. This section explains which parts of this document various types of readers should review.

**Types of Reader:**

This document has been provided for the following types of readers to facilitate clear understanding of the project requirements. The sections listed are crucial for the corresponding reader:

* Project Sponsor
* Project Manager
* Application Developers
* Database Administrators
* User Interface and Experience Designers
* Software Testers

**Technical Background Required:**

It is important to have a general understanding of various software systems and time or project management tools. Basic knowledge of servers and databases is an add on. Combining both will lead to more effective communication with the implementation team.

**Overview Sections:**

For an overall understanding of the project following sections should be read:

• Scope of the product

• Business case of the product

• Overview of the requirements document

• Product functions

**Reader-Specific Sections:**

The sections listed here are meant for specific readers and may be skipped by other readers:

**Application Developers:**

○ Specific Requirements

○ High-Level Technology Architecture

**Database Administrators:**

○ User Requirements

○ Reporting Requirements

○ Security Requirements

**User Interface Designers:**

○ Reporting Requirements

○ User Interface Requirements

**Software Testers:**

○ User Requirements

○ High-Level Technology Architecture

**Section Order Dependencies:**

The reader must have an understanding of the General Description before reading the Specific Requirements and High-Level Technology Architecture.

**Scope of the Product:**

Using the E-magnet tool will eliminate the manual filling to time sheets. It is integrated with the project management tool (JIRA). Any team member can create a task on JIRA and it will be directly imported to the E-magnet tool. When a developer hits the start task button, the time is automatically recorded till the pause or stop task button is hit. This time is then updated on JIRA and can be reviewed by the project manager for performance analysis. Also, after the developer pushes the code on GitHub, all the committed code will be visible to all the members. A personalized dashboard is provided to the developer, where he can rework on his code. Along with that, the tool provides a centralized platform, called the Calendar Management System, for the different members of a project team to know about each other’s work proceedings and time availability. The calendar management system, helps the team members to view each other’s available time slots, as this would help them in scheduling meetings and appointments.

The system maintains the time recorded for each member, to complete an assigned task. Also the system provides notification, when an activity has been completed.

**Business Case for the Product:**

This tool is used to remove the manual filling of time sheets and as a result saves a lot of time and money. This automation system will help in saving $3,75,000 annually. Also the other features provided by the tool help in getting more information on a single platform. In addition to the above capabilities, this product is efficient and it simplifies the process of creating tasks and scheduling appointments.

**Overview of the Requirements Document:**

Users can log into the E-magnet tool by creating a username and password

**1.2 Reference Materials:**

There are many other documents that together describe the complete set of requirements for the project.

Other documents such as standards, regulations, business process flows, or descriptions of existing functionality to be modified should be named here. Often there are copies of common information distributed throughout the company. The definitive source of the information and/or the person who manages the information should be noted so that discrepancies may be quickly resolved with clear authority.

| Reference Document Name | Brief Description | Location of Definitive Source |
| --- | --- | --- |
| Business Case Document | The business case document defines the existing problem and its impact on the Store. It is an overview of both the current and future state of our project | Google drive |

## 1.3 Specific Terms and Acronyms:

Terms here are specific to this document. Refer to “Project Glossary” for a more comprehensive list of terms used in this project.

Do not copy terms from the Glossary to this section. Describe the meaning and usage of any terms that are used in any of the requirements documents. Especially include a term if its meaning could be interpreted differently by members of the project team. If the company has a common definition of terms document or list of acronyms, it may be attached separately but the document must be referenced here.

| Term or Acronym | Description |
| --- | --- |
| JIRA | Project Management Tool |
| GitHub | Web based Git repository hosting service which offers all the distributed revision control and source code management |
| API | Application Program Interface |

## 1.4 User Roles:

Roles played by various users that interact with the business process or system is described here.

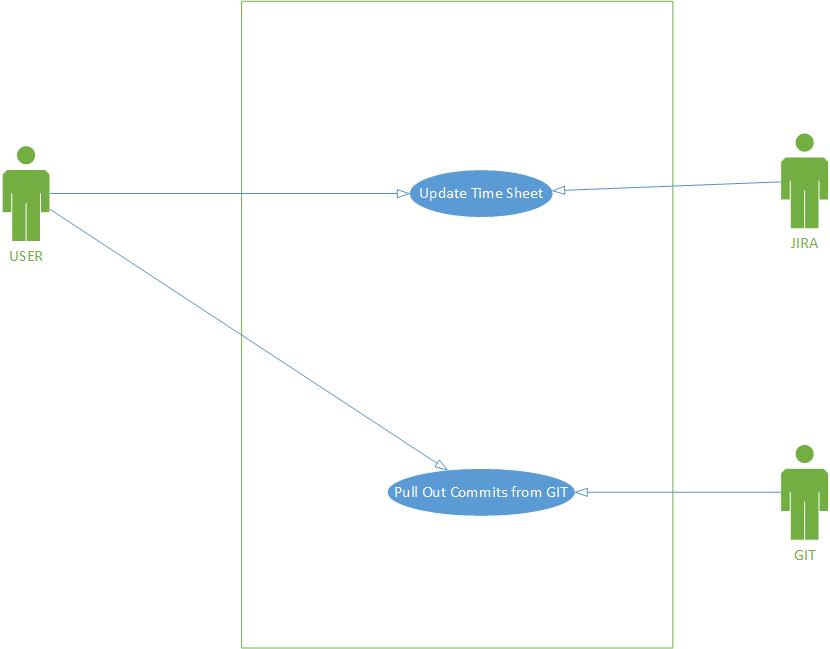
Name and describe the role of kind of user. If helpful to understanding, list job titles that might play each role. If there are special skills required, outline the steps to acquire the required knowledge. If authorization is required for certain operations, be sure to create a separate role for the authorized user. However, this section should not detail the user requirements. This section is still overview for the first-time reader.

| Role | Description of Role and Activities Performed |
| --- | --- |
| Admin | The person who creates accounts for other users and connects user accounts with GitHub and JIRA |
| Developer | The person who writes the code, commits the code, completes the tasks and uses personalized dashboard for re-work |
| General User | The team members who can access the tool to create tasks, view tasks and create appointments |
| Maintenance of the tool | The team who would look after tech support |

# Customer Workflows:

## As-Is Business Use Cases and Actors

The figure below depicts the ‘big picture’ of the business processes (business use cases) that will be modified or replaced with our E-Magnet system. The actors involved with each process are also shown.

**

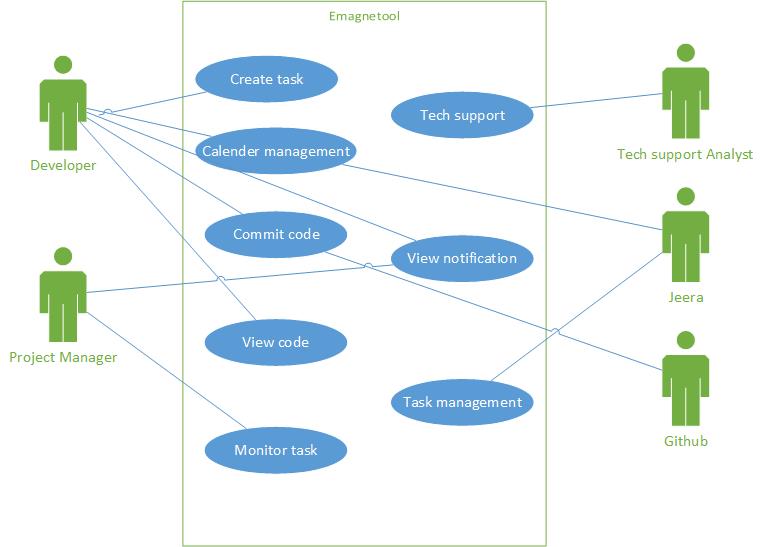
Business Use Case Diagram

Currently without our system being not implemented the user is going to waste ten minutes of time in filling the time sheets manually. He will be pulling out the commits from GitHub.

## 

## To-Be Use-Cases and Actors

The figure below shows all use cases and actors involved in our system E-Magnet.

**

Actors Generalization and Functional Requirements

| **Actor** | **Background and Skill Set** | **Goals** |
| --- | --- | --- |
| Developer | He is the one who writes the code and develops the system according to requirements. | Creating code, commit the task and pulling the commits from GitHub. |
| Project Manager | Allocating tasks, project management skills | Monitors the task, Views notification. |

| **Use-Case ID** | **Actor** | **Brief Description** |
| --- | --- | --- |
| U1 | Developer | Create the code and commits the code in GitHub. He makes appointments with other developers using calendar management. |
| U2 | Project Manager | Views notification about different commits of the developers. Monitors the task assigns new task to developers. |
| U3 | Tech Support Analyst | Provides technical support to the system |
| U4 | JIRA | It updates he task status as a result timesheet will be updated |
| U5 | GitHub | It is a repository where the code is committed and all the developers can pull the committed code. |

**General Description:**

**User Characteristics**

The members of an organization will be using the E-Magnet tool. Basic computer knowledge is sufficient to use the system for the users. Moreover, a tutorial will be provided for the first time users to guide them throughout the application process. Easy accessibility and timely management along with the simplicity of the system will motivate both publishers and advertisers to make use of it in the best way.

## 1.5 Assumptions

Assumption statements are not specific requirements themselves. They help clarify requirements that may be misinterpreted due to different definitions of terms or different opinions about how some business operation flows. During validation of requirements, assumptions may help discover omitted requirements, highlight project risks, or encourage discussion of various interpretations. Examples include: technology such as a specific operating system or database and expectations of the development team.

| ID | Assumption Statement | Related To |
| --- | --- | --- |
| A1 | We are assuming that we get the required API for GitHub and JIRA | Customer |
| A2 | We have authorization for fetching the data of the system | Related tools and customers |
| A3 | Customer should be using JIRA as a Project Management tool before using our tool, so that it integrates with our system | Customer |

## 1.6 Constraints:

Identify anything that puts limits on implementing the requirements.

This section is often misunderstood. Pay particular attention to this description. A constraint is a statement that expresses measurable bounds on the implementation of a system. It may limit the technology or environment in which the system may be built. It may list the certain options that may be considered for implementation. It may describe limitations or conditions that users must work under. In all cases, a “constraint” is differentiated from business, user and functional requirements in that it does not describe the functionality but rather limits the design or implementation options. Examples of constraints include: hardware constraints because the software may reside on a server with other applications and compete for resources such as memory, ports, and processor speed; reliability constraints such as how often the system must be backed up or even if hot back such as RAID is required; and “criticality” refers to how long the system would be allowed to be out of operation due to some failure. These are just a few of the constraints that may make the project a success.

| ID | Constraint Statement | Related To |
| --- | --- | --- |
| C1 | Internet connectivity and complying device to host the system needs to be essential for the system to work smoothly | Connectivity |
| C2 | Overlay system of the organization | Tool |
| C3 | If the tool or connecting software systems are under maintenance | Tool |
| C4 | Dependency on API’s for connecting tools | Tool |

## 

## 1.7 Dependencies:

Detail any external event, condition, or system that must be in place for a requirement to be valid.

| ID | Dependency Statement | Related To |
| --- | --- | --- |
| D1 | Dependent on the API’s from overlay tools to fetch data from their database | Internal condition |

# 2. Requirements:

## 2.1. Business Requirements

Business-level requirements are written from the sponsor’s perspective. The business requirements identify the reason why the project is being done or what business objective it supports, as well as the benefits to the business. Business requirements are typically documented early in the project life cycle or the planning phase of the project, and are frequently documented in the project management deliverables.

|  |  |
| --- | --- |
| ID | Business Requirement Statement |
| B1 | We require to cut down on the time spent of manual filling of time sheets and hence this tool was launched. This tool eases the work for an employee and directly records the time automatically. This results in more productivity and saves money too. The additional features of the tool also provides many functionalities which helps the employees to perform tasks.  We are the primary stakeholder of the product. Various companies will use our tool and benefit from the end product. |

## 

## 2.2. User and Functional Requirements

They are the Inputs and Outputs of the business process. They describe “what” the user expects to obtain or needs from the system. These statements support the business requirements. The Use Case form may also be used as a supplement to the User Requirements section.

| ID | User and Functional Requirement Statements | BR | CI | ST |
| --- | --- | --- | --- | --- |
| **User Role** | **Tool** |  |  |  |
| U1.1 | Tool Functions |  |  |  |
| **U1.1F1** | The tool should be able to add or remove member and projects. |  |  |  |
| **U1.1F2** | The tool should provide notification, calendar management and dashboard system. |  |  |  |
| **U1.1F3** | The tool should integrate well with JIRA |  |  |  |
| **User Role** | **Developer** |  |  |  |
| **U1.2F1** | Developer should be able to do his task and start/stop time for updating time sheets |  |  |  |
| **U1.2F2** | Developer should be able to view his committed code on the dashboard |  |  |  |

## Nonfunctional Requirements

Nonfunctional requirements focus on the qualities that must be applied to design and implement the system. These are specific standards and attributes in support of the other requirements. For detailed information about nonfunctional requirements, including over 2,000 suggested elicitation questions, reference   
The Quest for Software Requirements, by Roxanne E. Miller, [www.RequirementsQuest.com](http://www.RequirementsQuest.com).

Column Header Key: BR = Business Rules Identifier, CI = Common Information Identifier, ST = Status.

Status Column Key: A = Accepted, C = Changed since last review, N (or Blank) = New since last review.

| ID | Nonfunctional Requirement Statements | BR | CI | ST |
| --- | --- | --- | --- | --- |
| OPERATION Requirements: How well does the system perform for daily use? √ A | | | | |
| Access Security How well is the system guarded against unauthorized access? The extent to which the system is safeguarded against deliberate and intrusive faults from internal and external sources. | | | | |
| N-ACS1 |  |  | √ |  |
| N-ACS2 |  |  | √ |  |
| Availability How dependable is the system during normal operating times? The degree to which users can depend on the system to be up (able to function) during “normal operating times.” | | | | |
| N-AVL1 |  |  | √ | C |
| N-AVL2 |  |  | √ | C |
| Efficiency How fast can it process? How many can be processed? How well does the system respond? The extent to which the software system handles capacity, throughput, and response time. | | | | |
| N-EFC1 |  |  | √ | C |
| N-EFC2 |  |  | √ | C |
| Integrity How accurate and authentic are the data? The degree to which the data maintained by the software system are accurate, authentic, and without corruption. | | | | |
| N-INT1 |  | √ |  | C |
| N-INT2 |  | √ |  | C |
| Reliability How immune is the system to failure? The extent to which the software system consistently performs the specified functions without failure. | | | | |
| N-REL1 |  |  | √ | A |
| N-REL2 |  |  | √ | A |
| Survivability How resilient is the system from failure? The extent to which the software system continues to function and recovers in the presence of a system failure. | | | | |
| N-SRV1 |  |  | √ | C |
| N-SRV2 |  |  | √ | C |
| Usability How easy is it to learn and operate the system? The ease with which the user is able to learn, operate, prepare inputs, and interpret outputs through interaction with a system. | | | | |
| N-USE1 |  | √ |  |  |
| N-USE2 |  | √ |  |  |
| REVISION Requirements: How easy is it to correct errors and add functions? | | | | |
| Flexibility How easy is it to modify to work in different environments? The ease with which the software can be modified to adapt to different environments, configurations, and user expectations. | | | | |
| N-FLX1 |  | √ |  | C |
| N-FLX2 |  | √ |  | C |
| Maintainability How easy is it to upkeep and repair the system? The ease with which faults in a software system can be found and fixed. | | | | |
| N-MNT1 |  | √ |  |  |
| N-MNT2 |  | √ |  |  |
| Scalability How easy is it to expand or upgrade the system’s capabilities? The degree in which the system is able to expand its processing capabilities upward and outward to support business growth. | | | | |
| N-SCL1 |  |  | √ |  |
| N-SCL2 |  |  | √ |  |
| Verifiability How easy is it to show the system performs its functions? The extent to which tests, analysis, and demonstrations are needed to prove that the system will function as intended. | | | | |
| N-VER1 |  |  | √ | A |
| N-VER2 |  |  | √ | A |
| TRANSITION Requirements: How easy is it to adapt to changes in the technical environment? | | | | |
| Interoperability How easy is it to interface with another system? The extent to which the software system is able to couple or facilitate the interface with other systems. | | | | |
| N-IOP1 |  |  | √ | A |
| N-IOP2 |  |  | √ | A |
| Portability How easy is it to transport? The ease with which a software system can be transferred from its current hardware or software environment to another. | | | | |
| N-POR1 |  | √ |  | C |
| N-POR2 |  | √ |  | C |
| Reusability How easy is it to convert for use in another system? The extent to which a portion of the software system can be converted for use in another. | | | | |
| N-REU1 |  |  | √ | A |
| N-REU2 |  |  | √ | A |

## Reporting Requirements:

E-Magnet Tool should maintain the committed codes. The tool should also be well integrated with JIRA, so that after a task is completed it is directly updated on JIRA, on hitting the stop button. The tool also sends email notifications to all the members, on every task completed by the developer. The tool should also send monthly reports to the project manager.

## System and Integration Requirements:

The system should support various project management tools like JIRA.

Security Requirements:-

Adequate system security should be maintained to ensure privacy of history records. Admin should validate user details at the time of registration.

**Appendices**

## Revision History

Each time a change is made to the document (even if several are released together) make a short note describing the change briefly. Do not use non-descript statements like: “Modified issues” or “Changed some assumptions”. Rather, make statements like: “added issues 21 through 27, resolved issues 8, 17, 18.” To determine what version a change was released, review the “save date” of the file on the title page. Verify that all participants are using the same version of this document.

|  |  |  |  |
| --- | --- | --- | --- |
| Change Date | Changed by | Description of Change | Version |
| 02/16/2015 | Priybrata P | Initial release | 1.0 |
| 02/26/2015 | Kshitij K | Initial release | 2.0 |
| 03/05/2015 | Pooja G | Initial release | 3.0 |
| 03/19/2015 | Apoorva J | Initial release | 4.0 |
| 03/27/2015 | Sandeep B | Final release | 5.0 |
| 04/10/2015 | Umeaz K | Delivered | 6.0 |