**Software Requirements Specification**

**for**

**<**CIEvents**>**

**Version 1.0 approved**

**Prepared by <Roberto Cervantes, Daniela Miles, Brian Ortiz, and Brian Radomski>**

**<organization>**

**<September 23rd, 2015>**

**Table of Contents**

**Table of Contents**

**Revision History**

**1.** **Introduction**

1.1 Purpose

1.2 References

**2.** **Overall Description**

2.1 User Classes and Characteristics

2.2 Operating Environment

2.3 Design and Implementation Constraints

2.4 Assumptions and Dependencies

**3.** **External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

**4.** **System Use Cases**

4.1 Use case name and identifier

4.2 Create Event (U2)

4.3 Check Events (U3)

**5.** **Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

**6.** **Other Requirements**

**7.** **System Requirements Chart**

**Appendix A: Analysis Models**

**Appendix B: To Be Determined List**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Person’s Name, not “Company X”. Multiple people are okay if they all worked on this version |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

CIEvents is a web-based application where users are able to create and manage events in various locations such as CSUCI main campus, Santa Rosa Island etc., and invite others. Info about time and place of the event should be provided online. The system will alert the event creator X days before the event day in case of weather condition. Change and suggest the closest day with suitable weather. All event participants should be notified on case of weather forecast change vie email. Invited users should be able to decline or accept the invitation.

## **References**

*None*

# **Overall Description**

## **User Classes and Characteristics**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

User Classes:

1. event creator - Creates events and is able to invite regular users to events and manage those events. Event creators are notified in case of weather changes before the events in case they want to change the event details.
2. regular - Can search events, join events and can create an event to become an event creator.

## **Operating Environment**

As a web-based application, this product will support usage from Google Chrome, Mozilla Firefox, Safari, and Internet Explorer.

## **Design and Implementation Constraints**

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>*

*You may have some or None. Think about it though, if you write “none”, but it is obvious you should have some, I will deduct points!*

Due to program installation limitations, the amount of programs we can use in the implementation will be limited.

## **Assumptions and Dependencies**

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

None for now.

# **External Interface Requirements**

**SKIP ALL OF SECTION 3 for the SRS… We’ll do this as part of the GUI Prototype**

## **User Interfaces**

*<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>*

*Think about this in terms of standards, not specific features. For example,*

* *all buttons will have a black border*
* *all fonts will be Arial*
* *Draw a screen template showing “main area”, “menu here”, “status bar” and describe each component. If you have multiple screen layouts depending on the user’s current task/settings, describe them*
* *What screen resolutions will you support?*
* *Will you be Section 508 compliant? Are there any other standards you support?*
* *etc…*

## **Hardware Interfaces**

*<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>*

*If you system doesn’t include hardware, then you’ll have none. If it has hardware components, then you should describe (at a high level) how you interface with that hardware.*

## **Software Interfaces**

*<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>*

*These are internal connections to things like databases, web servers. You mainly need to explain that you have them, but (for CS421) I don’t expect detailed information about how you actually connect to them and use them. Just explain that you* ***do*** *connect to them and use them for storage of customer information, or to process incoming web requests, etc… These are internal components of your system.*

## **Communications Interfaces**

*<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>*

*These are external communication mechanisms. Do you connect to a bank computer to verify credit card information? That is NOT part of your system, so it is an external communication you have. Describe it here. Do you have other systems connecting in to yours to perform some function? That would also go here.*

# **System Use Cases**

*The overall use case diagram should be here.*

*The text description of each use case should follow.*

The application provides CI event organizers a means to create and manage events - set dates and times, invite users, and monitor conditions for the event such as weather. User should be able to edit and update the events at their discretion, updating even participants as required.

## **Use case name and identifier**

1. **Unique Identifier**
2. **Objective** - What is the ultimate objective of the use-case. What is it trying to achieve? What was the source of the use-case requirement?
3. **Priority** – The overall priority of this use-case (Low, Medium, High)
4. **Source** – Who is the main source of this use case. Who cares most about this functionality? This should be the one person you would ask if there is a question about this use-case. (Make up a name and cite their: John Smith (End-user) here.)
5. **Actors** - Who is involved in the use-case? Which actors/stakeholders?
6. **Flow of Events** 
   1. **Basic Flow** - flow of events normally executed in the use-case
   2. **Alternative Flow(s)** - a secondary flow of events due to infrequent conditions
   3. **Exception Flow(s)** - Exceptions that may happen during the execution of the use case
7. **Includes** - other use case IDs that are referenced in steps in the flow of events.
8. **Preconditions** - Any condition that must be satisfied before the use case begins. If the condition is “User is logged in”, then the first step of the use case is NOT “User logs in”. They are already logged in if that is a pre-condition!
9. **Post conditions** - The conditions that will be satisfied after the use case successfully completes
10. **Notes/Issues** - Any relevant notes or issues that need to be resolved

## Create Events**(U2)**

1. **U2**
2. **Objective** – The user is creating an event and will invite other users.
3. **Priority** – High
4. **Source** – Carl Gnome (marketing)
5. **Actors** – Event Creator, System
6. **Flow of Events** 
   1. **Basic Flow**
      1. User creates the event
      2. User confirms the event details
      3. System validates the event details
      4. User invites other users to the event
   2. **Alternative Flow 1** – The event is set to a past date
      1. An error message is displayed telling they need a date in he future
      2. Return to step 5.1.2
   3. **Alternative Flow 2** – At any step the user presses “cancel”
      1. System returns to the main menu
   4. **Exception Flow 1** –
      1. Database is locked due to backup in progress. System executes use case U5
7. **Includes**
   1. U5 – Exception occurs
8. **Preconditions** – User is logged in
9. **Post conditions** – Event has been created for user
10. **Notes/Issues** - None

## Check Events **(U3)**

0.

1. **U3**
2. **Objective-** The user checks the events and joins whichever ones he wishes
3. **Priority -** High
4. **Source -** Carl Gnome (marketing)
5. **Actors** - Regular, System
6. **Flow of Events**
   1. **Basic Flow**
      1. User checks available events
      2. User joins the event
      3. User confirms joining the event
      4. The system adds the user to the list of participants

6.2. **Alternative Flow 1** - The event is full

6.2.1 Error message is displayed telling regular the event is full

6.2.2 Returns to main event window

6.3 **Alternative Flow 2 -** At any step, the user presses “cancel”

6.3.1 System returns to main menu

7. **Includes**

8. **Preconditions -** User is logged in

9. **Post conditions -** User has joined an event

10.  **Notes/Issues -** None

# **Other Nonfunctional Requirements**

## **Performance Requirements**

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

***In this section, just say “See section 7 requirements 23-27”. And I’ll assume those requirements are Performance related.***

See section 7 requirements 001-005

## **Safety Requirements**

*<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>*

***In this section, just say “See section 7 requirements 25-32”. And I’ll assume those requirements are Safety related.***

## **Security Requirements**

*<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>*

***In this section, just say “See section 7 requirements 35-42”. And I’ll assume those requirements are Security related.***

## **Software Quality Attributes**

*<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>*

***In this section, just say “See section 7 requirements 55-62”. And I’ll assume those requirements are Software Quality related.***

# **Other Requirements**

# **System Requirements Chart**

*< Include a* ***table*** *in this section with the following columns:*

***ID*** *– Unique requirement ID*

***Priority*** *– Priority of this requirement*

***Type*** *– Functional(F) or Non-functional(NF)*

***Source*** *– Who is most interested in this requirement (John Smith – Customer). For this project you can make it up, in reality you’ll want to capture this as you capture the requirements.*

***Contained in Use Case(s****) – Which use cases reference this requirement or which use cases when executed will perform this requirement. There may be a few functional requirements without a use-case and the non-functional requirements generally will NOT be part of a use-case (so put N/A).*

***Description*** *– The description of the requirement. “The system shall …. “*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | *Priority* | *Type* | *Source* | *Contained in Use Case* | *Description* |
| *001* | *1* | *F* | *Brian Ortiz and Roberto Cervantes!-Customer* | *Event Creator* | *Email creator in case of weather changes* |
| *002* | *2* | *F* | *Roberto Cervantes and Brian Ortiz-Customer* | *System* | *Check for weather changes one week before the event.* |
| *003* | *3* | *F* | *Roberto Cervantes and Brian Ortiz-Customer* | *System, Event Creator* | *Strip the event creator of his title of event creator after event ends unless they ahve another event created* |
| *004* | *4* | *F* | *Roberto Cervantes and Brian Ortiz-Customer* | *Event Creator* | *Needs to provide approval from CSUCI* |
| *005* | *5* | *NF* | *Roberto Cervantes and Brian Ortiz-Customer* | *System* | *Limit locations to CI campuses* |

**Appendix A: Analysis Models**

**Appendix B: To Be Determined List**

*<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>*

*List here any open questions or things you know still need to be done to the SRS, but haven’t been addressed yet. (It’s okay to have things like that, especially in this CS421 project because we don’t have time to do everything.)*

What is a SRS, anyway? Also, perhaps there are a number of features that can implemented once the core functionality has been developed and settled - features such as improved GUI with integrated Google Maps and Google Weather support, and so on. Additional features to be developed on a tentative basis.