JMG Student Site Check-in Application

System Requirements Specification

Client
Lanet Anthony, Samantha Brink, JMG

Developer

Elijah Caret, Michael Ferris, Xingzhou Luo,
Spencer Morse, Brennan Schatzabel

University of Maine
October 25, 2021
Version 1.0

JMG Student Site Check-In Application
System Requirements Specification

Table of Contents

I. Introduction

The JMG Student Site Check-In Application is a service that aims to automate the process of JMG students to notify teachers of their attendance at an event outside of school that is counted towards course credit. This is a capstone project for Elijah Caret, Michael Ferris, Xingzhou Luo, Spencer Morse, and Brennan Schatzabel in partial fulfillment of the Computer Science B.S. degree for the University of Maine.

1. Purpose of the Document

The purpose of this document is to lay out the requirements for a student check-in application for the clients (Lanet Anthony and Samantha Brink from JMG) created by the development team (Cyber Cookie). It serves as a complete overview of exactly what the application will do upon completion.

This document contains functional and non-functional requirements, the tests that will be performed to ensure that the requirements are met, a list of deliverables, and any remaining open issues

2. Purpose of the Product

One of the responsibilities of the JMG program is to track and report the attendance of students at out-of-school positions, such as internships. JMG must be able to report this information back to the students' school and the out-of-school position needs to be able to confirm the attendance reported by the student. The JMG's LMS (learning management system) does not provide this functionality however, creating a problem for the JMG.

In order for the JMG to meet its responsibilities they will need an application that can keep track of self-reported student attendance, allow for an out-of-school overseer to rate their participation, and be visible to both teachers and JMG admins. This application will need to be web based and accessible via desktop and mobile devices. Additionally, the delivered product must interface with the salesforce API and preferably also the Learn Upon LMS API.

3. Product Scope

The system will be a web application. Included is the backend software (Python or other language) to help with making database queries, interpreting user actions, as well as sending and accessing data from Salesforce. Another piece is the front end, which will encode the user interface and look of the web application, this also how the user will

interact with the system. Finally, there will be a database (SQL or some relational database package) that will hold data such as login information.

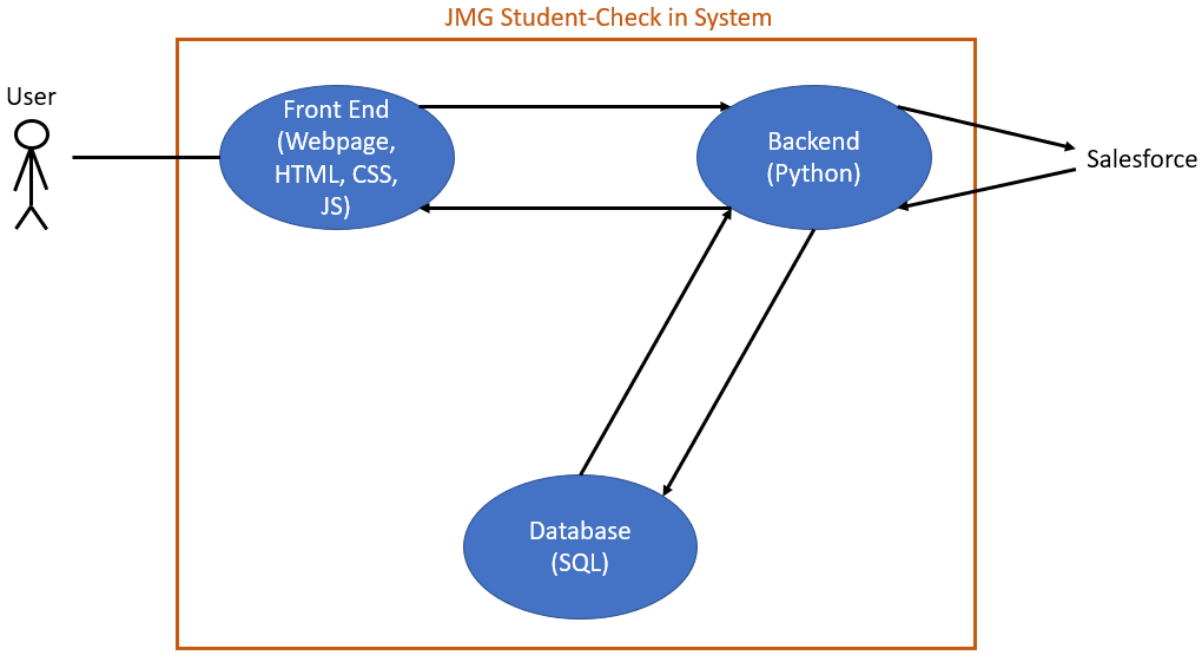


Figure I.3: product scope overview

4. Reference

Salesforce. (n.d.). *REST API Developer Guide*. Developer Portal. Retrieved October 26, 2021, from https://developer.salesforce.com/docs/atlas.en-us.api_rest.meta/api_rest/intro_rest.htm.

Martin, M. (2021, October 6). *Functional requirements vs non-functional requirements: Differences*. Guru99. Retrieved October 26, 2021, from https://www.guru99.com/functional-vs-non-functional-requirements.html.

US Department of Education (ED). (2021, August 25). *Family educational rights and privacy act (FERPA)*. Home. Retrieved October 25, 2021, from https://www2.ed.gov/policy/gen/guid/fpco/ferpa/index.html

II. Functional Requirements

Based on the proposals and proposal review meeting with the client. These functional requirements were developed to encapsulate the basic functionality of the system.

1. The user shall be able to create a new account with a credential combination.
2. The user shall log in to the system with a credential combination.
3. The system shall verify the user's credential combination when logging in.
4. The system shall exchange data with Salesforce via the REST API.
5. No user shall be able to view another user's data except for the system administrator.
6. The system shall allow the student user to check in with a button click.
7. The system shall notify the student user whether the check-in was successful.
8. The system shall notify the site supervisor when a student user has checked in.
9. The system shall allow the site supervisors to fill out a form to provide feedback on student performance.
10. The system shall allow the school supervisor to view the site supervisor's feedback for its students.
11. The system shall allow the school supervisor to check student users in when they forget to check in themselves.
12. The system shall notify the school supervisor whether a student's attempt to check in was successful.
13. The system shall allow the school supervisor to view their students' site visits

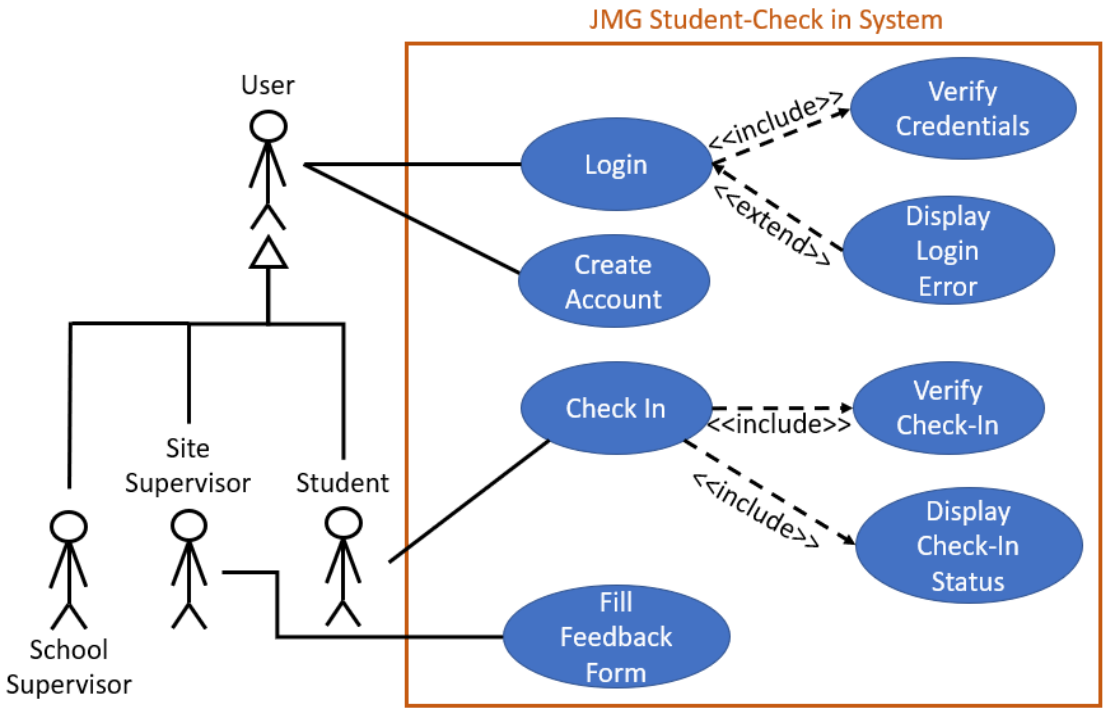


Figure II.a: use case diagram for use cases 1 - 3, and 6 - 9

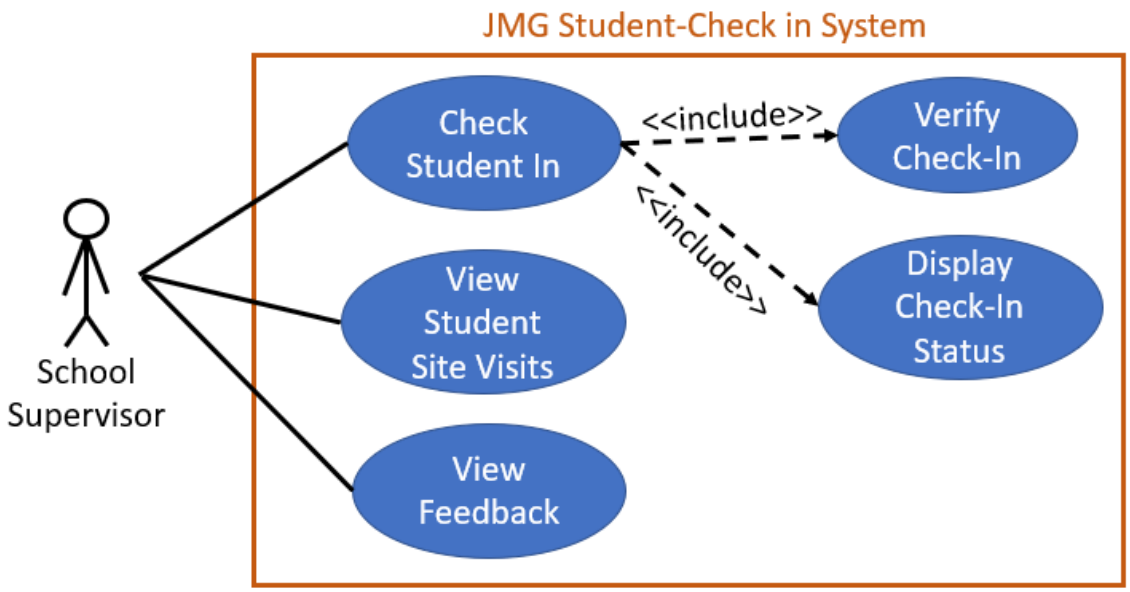


Figure II.b: use case diagram for use cases 10 - 13

Based on the functional requirements, most use cases are described using use case specifications.

Number	1	
Name	Log in	
Summary	User logs in by entering a credential combination	
Priority	5	
Preconditions	N/A	
Postconditions	User is viewing main check-in screen	
Primary Actor	User	
Secondary Actors	N/A	
Trigger	User opens check-in application website	
Main Scenario	**Step**	**Action**
	1	System displays login screen
	2	User enters a credential combination
	3	System verifies the credential combination
Extensions	**Step**	**Branching Action**
	3a	<password invalid>**:** <system displays small text box notifying user that login failed>
Open Issues	N/A	

Number	2	
Name	Check in	
Summary	Student checks in the site	
Priority	5	
Preconditions	User successfully logged in	
Postconditions		
Primary Actor	Student	
Secondary Actors	N/A	
Trigger	N/A	
Main Scenario	**Step**	**Action**
	1	User clicks “check-in” button
	2	System verifies check-in
	3	System displays text box notifying user that check-in was successful
Extensions	**Step**	**Branching Action**
	2a	<current time outside of site visit time>**:** <system displays text box notifying user that check-in failed>
Open Issues	Not sure how to approach unexpected check-ins	

Number	3	
Name	Fill Feedback Form	
Summary	Site supervisor fills out form assessing student performance for session	
Priority	4	
Preconditions	Student is checked in to site, site supervisor has received notification	
Postconditions	Submitted form should be visible to school supervisor	
Primary Actor	Site supervisor	
Secondary Actors		
Trigger	Site supervisor opens feedback form	
Main Scenario	**Step**	**Action**
	1	Site supervisor fills out feedback form (text box, checkboxes, etc.)
	2	Site supervisor clicks submit button
	3	system displays message notifying site supervisor that the form was submitted successfully
Extensions	**Step**	**Branching Action**
	3a	<form not completed>**:** <system reloads form highlighting missing information>
Open Issues	NA	

Number	4	
Name	Check Student In	
Summary	School supervisor checks student in when student forgot to check in	
Priority	4	
Preconditions	Student has not checked in to site	
Postconditions	Student is fully checked in to site	
Primary Actor	School Supervisor	
Secondary Actors		
Trigger	School supervisor notified that student was at site-by-site supervisor	
Main Scenario	**Step**	**Action**
	1	School supervisor open's student site visit information
	2	School supervisor check's student in
	3	System displays text box notifying school supervisor that student was successfully checked in
Extensions	**Step**	**Branching Action**
	3a	<check in fails>**:** <system displays text box notifying school supervisor that attempt to check in has failed>
Open Issues	NA	

Number	5	
Name	View Student Site Visits	
Summary	School supervisor views all of the site visits of a particular student	
Priority	3	
Preconditions	School supervisor is logged in	
Postconditions	NA	
Primary Actor	School Supervisor	
Secondary Actors	NA	
Trigger	NA	
Main Scenario	**Step**	
	1	School supervisor clicks on certain student they want info on
	2	System displays options
	3	School supervisor clicks on "View check-in history"
	4	System loads page showing student's check-in history
Open Issues	NA	

Number	6	
Name	View Feedback	
Summary	School supervisor views all performance assessments of student	
Priority	5	
Preconditions	School supervisor is logged in	
Postconditions	NA	
Primary Actor	School Supervisor	
Secondary Actors	NA	
Trigger	NA	
Main Scenario	**Step**	
	1	School supervisor clicks on certain student they want info on
	2	System displays options
	3	School supervisor clicks on “View performance history”
	4	System loads page showing student’s performance assessments
Open Issues	NA	

Number	7	
Name	Create Account	
Summary	New user creates a new account	
Priority	3	
Preconditions	User is new	
Postconditions	User is now registered in system	
Primary Actor	User	
Secondary Actors	NA	
Trigger	NA	
Main Scenario	**Step**	
	1	User clicks on create account button
	2	System displays text fields for username, password, and email
	3	User enters username, password, and email
	4	User clicks “create account” button
	5	System verifies information
	6	System sends confirmation email to user
Extensions	**Step**	**Branching Action**
	6a	<username/password/email already in system> <system displays message telling user that this username/email/password is already in use>
Open Issues	NA	

We envision the testing of these requirements through a demonstration of the application and displaying each piece of functionality to ensure that the requirements are met. A fake account for a student, school supervisor, and site supervisor will be made for the sake of the demonstration. First a login attempt will be made with an incorrect username and

password, followed by the correct username and password. The student account dashboard will be shown, with demonstrations of successfully and unsuccessfully checking in to a site. The site supervisor account dashboard will be shown, with a demonstration of the student assessment form including premature submission. The school supervisor dashboard will be shown, demonstrating their ability to see all their students check ins, as well as the assessments for each session.

III. Non-Functional Requirements

Based on the proposals and proposal review meeting with the client. These non-functional requirements were developed to define the basic performance of the system.

1. The system shall be able to handle 1000 users without impacting its performance. (Lowest Priority)
2. A data request or transmission to the Salesforce system shall take no longer than 10 seconds. (Low Priority)
3. The system should respond to any user request within at least 10 seconds. (High Priority)
4. The system shall perform any database query within at least 10 seconds. (Medium Priority)
5. The system shall follow FERPA guidelines. (Highest Priority)
6. Any new web page must be loaded within 10 seconds of the user initiating it. (High priority)
7. The application should adapt to any size screen including mobile. (Highest Priority)
8. The application should load within at least 10 seconds when there are less than 1000 concurrent users. (High Priority)
9. The application should load within at least 20 seconds when there are more than 1000 concurrent users. (Lowest Priority)
10. The system shall record any failed login attempts. (Low Priority)

IV. User Interface

See "*User Interface Design Document*" for JMG Student Check-in Application.

V. Deliverables

The software should all be available to the clients via the shared GitHub repository including the source code, executables, as well as additional documentations. The source code will include:

- JavaScript, HTML, and CSS files for front end and user interface
- Either Python/Ruby scripts to build backend from scratch
- Use of Budibase as a potential low-code platform

For documentation, the repository will eventually have:

- System Requirements Specification (SRS) (October 25th)
- System Design Document (SDD) (November 10th)
- User Interface Design Document (UIDD) (November 29th)
- Critical Design Review Document (CDRD) (December 13th)
- Code Inspection Report (CIR) (2nd Semester)
- User Manual (UM) (2nd Semester)
- Administrators Manual (AM) (2nd Semester)
- Final Project Report (FPR) (2nd Semester)
- Biweekly Status Reports

There will also be hard copies made of these documents.

Some other documentation that may be included are:

- Sequence Diagrams
- Use Case Models
- Testing Plans (Unit, Regression, Acceptance, Integration, etc.)

VI. Open Issues

These issues will be addressed later in the development process and are as follows:

- Process of determining when a student can no longer check themselves in, especially during unexpected site visits.

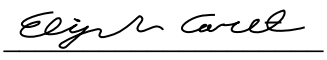
Appendix A: Agreement between Customer and Contractor

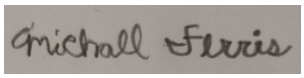
By signing this document, all parties agree that this is a complete list of requirements and deliverables for the JMG Student Check-in Site application, apart from the current open issues (addressed in Section VI). The tests described in this document (under sections II and III) will be the metric by which the product is deemed complete. Once all

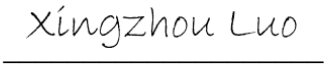
requirements are complete and verified by said tests, the application will be finished and ready for use by JMG.

In the case that requirements or any other information in this document need to change for the contract to be fulfilled, the following procedure will be followed: The party that believes a change is necessary shall contact the other party, explaining the situation. A meeting between the two parties will be held to discuss the problem and possible solutions. Once an agreement has been reached, modifications to this document will be made to reflect it, and all members of each party will sign the new document, which will then replace this one.

Team Members:

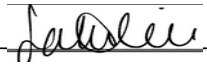
Name: Elijah Caret Signature: 

Name: Michael Ferris Signature: 

Name: Xingzhou Luo Signature: 

Name: Spencer Morse Signature: 

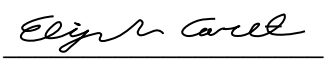
Customers:

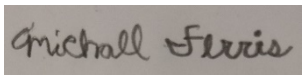
Name: Samantha Brink Signature: 

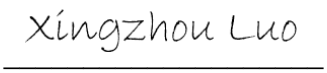
Appendix B: Team Review Sign-off

By signing below, all members agree that they have reviewed this document and agree on its content and format.

Team Members:

Name: Elijah Caret Signature:  Date: Oct. 25, 2021

Name: Michael Ferris Signature:  Date: Oct. 25, 2021

Name: Xingzhou Luo Signature:  Date: Oct. 25, 2021

Name: Spencer Morse Signature: Spencer Morse Date: Oct. 25, 2021

Appendix C: Document Contributions

- Elijah Caret:
  Introduction, Functional Requirements, Non-Functional Requirements, Deliverables, Open Issues, Appendices
- Michael Ferris:
  Introduction, Appendices
- Xingzhou Luo:
  Introduction, Functional Requirements, Non-Functional Requirements, Appendices
- Spencer Morse:
  Open Issues, Appendices
- Brennan Schatzabel:
  Introduction