# **CSC 431** Cadastre UI Software Requirements Specification (SRS)

**Team 4 : !bad**

|  |  |
| --- | --- |
| Amanda Davis | Team Member |
| Halle Fields | Team Member |
| Nathan Fox | Team Member |
| Devin Michaels | Team Member |
| Brandon Purvis | Team Member |
| Alex Voutsinas | Team Member |

# Version History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Author(s) | Change Comments |
| 0.0.1 | 2/28/2018 | Davis, Fields, Fox, Purvis, Michaels, Voutsinas | Created document, assigned team roles, wrote initial requirements, use-case diagrams |
| 1.0.0 | 3/05/2018 | Voutsinas | Finalized first draft |
| 1.0.1 | 5/04/2018 | Fields | Updated to current scope, made presentable |
| 1.1.0 | 5/07/2018 | Davis, Fields, Fox, Purvis, Michaels, Voutsinas | Completed document for Spring 2018, presented to the class |

# Table of Contents

[1. System Requirements](#_3dy6vkm) 5

[1.1 Functional Requirements](#_1t3h5sf) 5

[1.1.1](#_4d34og8) Polygon Drawing 5

1.1.2 Polygon Neighbor Awareness 5

1.1.3New Polygons Snap to Existing Points 5

1.1.4 Collision Detection/Resolution 6

1.1.5 Storing Polygons in Database 6

1.1.6 Polygon Review/Approval Process 6

[1.2 Non-Functional Requirements](#_2s8eyo1) 6

[1.2.1](#_17dp8vu) Slippy Tiled Images 6

[1.2.2](#_17dp8vu) Point and Click Polygon Drawing 7

[2. System Constraints](#_3rdcrjn) 8

[2.1 Tool Constraints](#_26in1rg) 8

[2.1.1](#_lnxbz9) Leaflet.js 8

[2.2 Language Constraints](#_35nkun2) 8

[2.2.1](#_1ksv4uv) JavaScript Front End 8

[2.3 Platform Constraints](#_44sinio) 8

[2.3.1](#_2jxsxqh) geoJSON Geographic Format 8

[2.3.2 Web Interface Requirement](#_2jxsxqh) 8

[2.4 Hardware Constraints](#_z337ya) 9

[2.4.1](#_3j2qqm3) Cloud Server 9

[2.5 Network Constraints](#_1y810tw) 9

[2.5.1](#_4i7ojhp) None Provided 9

[2.6 Deployment Constraints](#_2xcytpi) 9

[2.6.1](#_1ci93xb) Integration 9

[2.7 Transition & Support Constraints](#_3whwml4) 9

[2.7.1](#_2bn6wsx) Easily Understandable 9

[2.7.2 No Support](#_2bn6wsx) 9

[2.8 Budget & Schedule Constraints](#_qsh70q) 9

[2.8.1](#_3as4poj) Semester-long Schedule 10

[3. Requirements Modeling](#_2p2csry) 11

[3.1.1](#_147n2zr) Use Cases 11

[4. Evolutionary Requirements (TBA)](#_3o7alnk) 12

[4.1 Functional Requirements](#_23ckvvd) 12

[4.1.1 Requirement Title](#_ihv636) 12

[4.2 Non-Functional Requirements](#_32hioqz) 12

[4.2.1 Requirement Title](#_1hmsyys) 12

# Table of Tables

[1. System Requirements](#_3dy6vkm) 5

[1.1 Functional Requirements](#_1t3h5sf) 5

[1.1.1](#_4d34og8) Polygon Drawing 5

1.1.2 Polygon Neighbor Awareness 5

1.1.3New Polygons Snap to Existing Points 5

1.1.4 Collision Detection/Resolution 6

1.1.5 Storing Polygons in Database 6

1.1.6 Polygon Review/Approval Process 6

[1.2 Non-Functional Requirements](#_2s8eyo1) 6

[1.2.1 Slippy Tiled Images](#_17dp8vu) 6

[1.2.2](#_17dp8vu) Point and Click Polygon Drawing 7

[2. System Constraints](#_3rdcrjn) 8

[2.1 Tool Constraints](#_26in1rg) 8

[2.1.1 Leaflet.js](#_lnxbz9) 8

[2.2 Language Constraints](#_35nkun2) 8

[2.2.1 JavaScript Front End](#_1ksv4uv) 8

[2.3 Platform Constraints](#_44sinio) 8

[2.3.1 geoJSON Geographic Format](#_2jxsxqh) 8

[2.3.2 Web Interface Requirement](#_2jxsxqh) 8

[2.4 Hardware Constraints](#_z337ya) 9

[2.4.1 Cloud Server](#_3j2qqm3) 9

[2.5 Network Constraints](#_1y810tw) 9

[2.5.1 None Provided](#_4i7ojhp) 9

[2.6 Deployment Constraints](#_2xcytpi) 9

[2.6.1 Integration](#_1ci93xb) 9

[2.7 Transition & Support Constraints](#_3whwml4) 9

[2.7.1 Easily Understandable](#_2bn6wsx) 9

[2.7.2 No Support](#_2bn6wsx) 9

[2.8 Budget & Schedule Constraints](#_qsh70q) 9

[2.8.1 Semester-long Schedule](#_3as4poj) 10

[4. Evolutionary Requirements (TBA)](#_3o7alnk) 12

[4.1 Functional Requirements](#_23ckvvd) 12

[4.1.1 Requirement Title](#_ihv636) 12

[4.2 Non-Functional Requirements](#_32hioqz) 12

[4.2.1 Requirement Title](#_1hmsyys) 12

# Table of Figures

[3. Requirements Modeling](#_2p2csry) 11

[3.1.1 Use Cases](#_147n2zr) 11

### System Requirements

#### Functional Requirements

##### Polygon Drawing

|  |  |
| --- | --- |
| Title | Polygon Drawing |
| Description | Users must be able to draw polygons on a map that are then stored as discrete properties. |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 0 |
| Precondition(s) | A vector layer drawing interface on top of a slippy composite image |
| Postconditions(s) | The polygon enters the approval process and is fed to the Collision Detection/Resolution. |
| Use Case Diagram | User to “Draws property boundaries” |

##### Polygon Neighbor Awareness

|  |  |
| --- | --- |
| Title | Polygon Neighbor Awareness |
| Description | Polygons need to know their relation to other polygons - which vertices they share, if they’re overlapping |
| Source Scenario |  |
| Priority | 0 |
| Precondition(s) | Polygon Drawing |
| Postconditions(s) | Ability to enter own property boundaries |
| Use Case Diagram | “Established polygons are aware of new polygon” extends “Polygon is approved” |

##### New Polygons Snap to Existing Points

|  |  |
| --- | --- |
| Title | New Polygons Snap to Existing Points |
| Description | Polygons should be able to snap into a correct format to a certain distance |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 1 |
| Precondition(s) | Polygon Drawing |
| Postconditions(s) | Creating accurate polygons is quicker and easier |
| Use Case Diagram | “Polygon snaps to existing points” extends “Draws property boundaries” |

##### 

##### Collision Detection/Resolution

|  |  |
| --- | --- |
| Title | Collision Detection/Resolution |
| Description | The program needs to recognize when two polygons are overlapping and find some way to deal with this issue. |
| Priority | 0 |
| Precondition(s) | Polygon Drawing |
| Postconditions(s) | We can now make drawn polygons legal if they pass this step |
| Use Case Diagram | Admin to “Overlapping/collision detected” |

##### Storing Polygons in Database

|  |  |
| --- | --- |
| Title | Storing Polygons in Database |
| Description | Legal, approved polygons need to be exported to the system database. |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 1 |
| Precondition(s) | Polygon Drawing; Polygon Review/Approval Process |
| Postconditions(s) | Export of Polygon Data Structure to Backend Database |
| Use Case Diagram | User to “Sends polygon for approval” to Admin to “Polygon is approved” |

##### Polygon Review/Approval Process

|  |  |
| --- | --- |
| Title | Polygon Review/Approval Process |
| Description | After polygons are drawn, they are not immediately stored as valid. They must meet polygon definition requirements and not overlap other valid polygons. |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 2 |
| Precondition(s) | Polygon Drawing |
| Postconditions(s) | Polygon Data Structure Ready to Store |
| Use Case Diagram | Admin to “Polygon is approved” to the User is “Notified about status of polygon” |

#### Non-Functional Requirements

##### Slippy Tiled Images

|  |  |
| --- | --- |
| Title | Slippy Tiled Images |
| Description | Need to be able to load images and drag them around a la google maps |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 2 |
| Applicable FR(s) | Storing polygons in external database (location of polygons changes as you drag the map) |

##### 

##### Point and Click Polygon Drawing

|  |  |
| --- | --- |
| Title | Point and Click Polygon Drawing |
| Description | Preferred method by which polygons are drawn. Clicks draw vertices and edges are automatically drawn between them. |
| Source Scenario | <Code for associated scenario in SCD> |
| Priority | 3 |
| Applicable FR(s) | Customizable snap |

### System Constraints

#### Tool Constraints

##### Leaflet.js

|  |  |
| --- | --- |
| Title | Leaflet.js |
| Description | Leaflet is the leading open- source JavaScript library for mobile-friendly interactive maps. |
| Priority | 0 |

#### Language Constraints

##### JavaScript Front End

|  |  |
| --- | --- |
| Title | JavaScript |
| Description | Front end needs to be written in JavaScript to run in the browser |
| Priority | 0 |

#### Platform Constraints

##### geoJSON Geographic Format

|  |  |
| --- | --- |
| Title | geoJSON |
| Description | All geographic data, figures, and figure collections must be encoded in the geoJSON format as described in RFC 7946. |
| Priority | 0 |

##### 

##### Web Interface Requirement

|  |  |
| --- | --- |
| Title | Web Interface |
| Description | Program must be accessible online as opposed to any static locations because internet cafes are the norm in Columbia as opposed to personal computers |
| Priority | 0 |

#### 

#### Hardware Constraints

##### Cloud Server

|  |  |
| --- | --- |
| Title | Cloud Server |
| Description | Program will be run on cloud servers not on local computer so hardware constraints will not be exceeded |
| Priority | 0 |

#### Network Constraints

##### None Provided

|  |  |
| --- | --- |
| Title | None Provided |
| Description | Networks constraints are determined by the systems that the web application will be running on, which have not yet been developed by its respective team. |
| Priority | N/A |

#### Deployment Constraints

##### Integration

|  |  |
| --- | --- |
| Title | Integration |
| Description | System must be able to fully integrate with the other 7 teams post-development. This will be done by an outside party. |
| Priority | N/A |

#### Transition & Support Constraints

##### Easily Understandable

|  |  |
| --- | --- |
| Title | Easily Understandable |
| Description | Able to be taken over by a new team upon the closing of the Spring 2018 semester. |
| Priority | 3 |

##### No Support Team

|  |  |
| --- | --- |
| Title | No Support Team |
| Description | No available support team, must be able to be self-maintained or have minimal support requirements. |
| Priority | 3 |

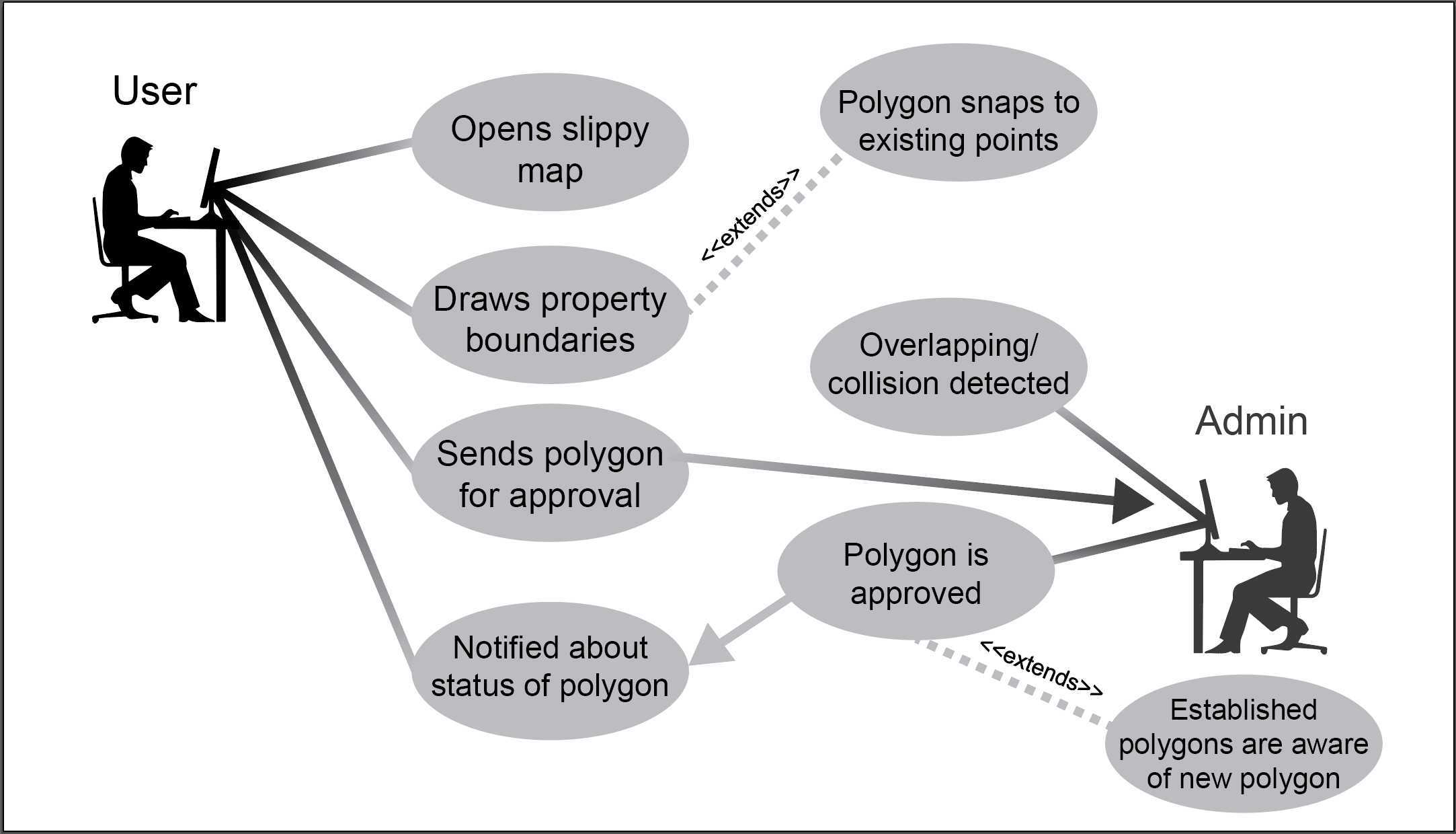
#### Budget & Schedule Constraints

##### Semester-long Schedule

|  |  |
| --- | --- |
| Title | Semester-long Schedule |
| Description | Schedule is limited to length of Spring 2018 Semester, which had 10 weeks remaining at project initialization. |
| Priority | 0 |

### Requirements Modeling

##### Use Cases



Use Case 1:

Customer opens a slippy map a la Google Maps and draws a polygon, denoting their property lines. Polygons are integrated to share points with existing polygons. Conflicts are detected and reported for resolution by an administrator.

Should stay general to the breakdown of what the customer experiences, not the details of how we resolve it.

Please update the table of figures to include any figures you generate as well as the Use Case entry for any of the above tables, try to link the names of the requirements/processes to the areas of the customer experience.

### Evolutionary Requirements (TBA)

1. At this time we do not have any identifiable evolutionary requirements.

#### Functional Requirements

*< List all functional requirements in the following example format >*

##### **Requirement Title**

|  |  |
| --- | --- |
| Title | <Insert title> |
| Description | <A one or two sentence description> |
| Priority | <Priority from 0 (highest) – 5 (lowest)> |
| Precondition(s) | <What needs to happen before> |
| Postconditions(s) | <What happens as a result> |
| Use Case Diagram | <Link or number, if present> |

##### 

|  |  |
| --- | --- |
| Title | <Insert title> |
| Description | <A one or two sentence description> |
| Priority | <Priority from 0 (highest) – 5 (lowest)> |
| Precondition(s) | <What needs to happen before> |
| Postconditions(s) | <What happens as a result> |
| Use Case Diagram | <Link or number, if present> |

#### Non-Functional Requirements

*< List all non-functional requirements in the following example format >*

##### **Requirement Title**

|  |  |
| --- | --- |
| Title | <Insert title> |
| Description | <A one or two sentence description> |
| Priority | <Priority from 0 (highest) – 5 (lowest)> |
| Applicable FR(s) | <Which functional requirement(s) is this applicable to?> |