**GPS Trail**

*-*

**Chris Lenell, Deonvell Shed,**

**Luke Wittenkeller, and Fade Abdeljaber**

**For use in CS 440**

**at the**

**University of Illinois Chicago**

**September 2018**

**Table of Contents**

[List of Figures 7](#_heading=h.gjdgxs)

[List of Tables 8](#_heading=h.30j0zll)

[I](#_heading=h.1fob9te) Project Description 9

[1](#_heading=h.3znysh7) Project Overview 9

[2](#_heading=h.2et92p0) The Purpose of the Project 9

[2a](#_heading=h.tyjcwt) The User Business or Background of the Project Effort 9

[2b](#_heading=h.3dy6vkm) Goals of the Project 9

[2c](#_heading=h.1t3h5sf) Measurement 9

[3](#_heading=h.4d34og8) The Scope of the Work 9

[3a](#_heading=h.2s8eyo1) The Current Situation 10

[3b](#_heading=h.17dp8vu) The Context of the Work 10

[3c](#_heading=h.3rdcrjn) Work Partitioning 11

[3d](#_heading=h.26in1rg) Competing Products 12

[4](#_heading=h.lnxbz9) The Scope of the Product 12

[4a](#_heading=h.35nkun2) Scenario Diagram(s) 12

[4b](#_heading=h.1ksv4uv) Product Scenario List 12

[4c](#_heading=h.44sinio) Individual Product Scenarios 12

[5](#_heading=h.2jxsxqh) Stakeholders 12

[5a](#_heading=h.z337ya) The Client 12

[5b](#_heading=h.3j2qqm3) The Customer 13

[5c](#_heading=h.1y810tw) Hands-On Users of the Product 13

[5d](#_heading=h.4i7ojhp) Maintenance Users and Service Technicians 13

[5e](#_heading=h.2xcytpi) Other Stakeholders 13

[5f](#_heading=h.1ci93xb) User Participation 13

[5g](#_heading=h.3whwml4) Priorities Assigned to Users 13

[6](#_heading=h.2bn6wsx) Mandated Constraints 14

[6a](#_heading=h.qsh70q) Solution Constraints 14

[6b](#_heading=h.3as4poj) Implementation Environment of the Current System 14

[6c](#_heading=h.1pxezwc) Partner or Collaborative Applications 14

[6d](#_heading=h.2p2csry) Off-the-Shelf Software 14

[6e](#_heading=h.147n2zr) Anticipated Workplace Environment 14

[6f](#_heading=h.3o7alnk) Schedule Constraints 14

[6g](#_heading=h.23ckvvd) Budget Constraints 15

[7](#_heading=h.ihv636) Naming Conventions and Definitions 15

[7a](#_heading=h.32hioqz) Definitions of Key Terms 15

[7b](#_heading=h.1hmsyys) UML and Other Notation Used in This Document 15

[7c](#_heading=h.41mghml) Data Dictionary for Any Included Models 15

[8](#_heading=h.2grqrue) Relevant Facts and Assumptions 15

[8a](#_heading=h.vx1227) Facts 15

[8b](#_heading=h.3fwokq0) Assumptions 15

[II](#_heading=h.1v1yuxt) Requirements 16

[9](#_heading=h.4f1mdlm) Product Use Cases 16

[9a](#_heading=h.2u6wntf) Use Case Diagrams 16

[9b](#_heading=h.28h4qwu) Product Use Case List 17

[9c](#_heading=h.nmf14n) Individual Product Use Cases 17

[10](#_heading=h.37m2jsg) Functional Requirements 19

[11](#_heading=h.46r0co2) Data Requirements 19

[12](#_heading=h.111kx3o) Performance Requirements 20

[12a](#_heading=h.3l18frh) Speed and Latency Requirements 20

[12b](#_heading=h.4k668n3) Precision or Accuracy Requirements 20

[12c](#_heading=h.1egqt2p) Capacity Requirements 20

[13](#_heading=h.2dlolyb) Dependability Requirements 21

[13a](#_heading=h.sqyw64) Reliability Requirements 21

[13b](#_heading=h.1rvwp1q) Availability Requirements 21

[13c](#_heading=h.2r0uhxc) Robustness or Fault-Tolerance Requirements 21

[13d](#_heading=h.3q5sasy) Safety-Critical Requirements 22

[14](#_heading=h.kgcv8k) Maintainability and Supportability Requirements 22

[14a](#_heading=h.34g0dwd) Maintenance Requirements 22

[14b](#_heading=h.43ky6rz) Supportability Requirements 22

[14c](#_heading=h.xvir7l) Adaptability Requirements 23

[14d](#_heading=h.1x0gk37) Scalability or Extensibility Requirements 23

[14e](#_heading=h.2w5ecyt) Longevity Requirements 23

[15](#_heading=h.3vac5uf) Security Requirements 23

[15a](#_heading=h.2afmg28) Access Requirements 24

[15b](#_heading=h.39kk8xu) Integrity Requirements 24

[15c](#_heading=h.48pi1tg) Privacy Requirements 24

[15d](#_heading=h.1302m92) Audit Requirements 24

[15e](#_heading=h.2250f4o) Immunity Requirements 25

[16](#_heading=h.319y80a) Usability and Humanity Requirements 25

[16a](#_heading=h.1gf8i83) Ease of Use Requirements 25

[16b](#_heading=h.2fk6b3p) Personalization and Internationalization Requirements 25

[16c](#_heading=h.3ep43zb) Learning Requirements 26

[16d](#_heading=h.4du1wux) Understandability and Politeness Requirements 26

[16e](#_heading=h.184mhaj) Accessibility Requirements 26

[16f](#_heading=h.279ka65) User Documentation Requirements 27

[16g](#_heading=h.36ei31r) Training Requirements 27

[17](#_heading=h.45jfvxd) Look and Feel Requirements 27

[17a](#_heading=h.2koq656) Appearance Requirements 27

[17b](#_heading=h.3jtnz0s) Style Requirements 28

[18](#_heading=h.4iylrwe) Operational and Environmental Requirements 28

[18a](#_heading=h.2y3w247) Expected Physical Environment 28

[18b](#_heading=h.3x8tuzt) Requirements for Interfacing with Adjacent Systems 28

[18c](#_heading=h.rjefff) Productization Requirements 29

[18d](#_heading=h.1qoc8b1) Release Requirements 29

[19](#_heading=h.2pta16n) Cultural and Political Requirements 29

[19a](#_heading=h.14ykbeg) Cultural Requirements 29

[19b](#_heading=h.243i4a2) Political Requirements 30

[20](#_heading=h.338fx5o) Legal Requirements 30

[20a](#_heading=h.1idq7dh) Compliance Requirements 30

[20b](#_heading=h.2hio093) Standards Requirements 30

[21](#_heading=h.3gnlt4p) Requirements Acceptance Tests 31

[21a](#_heading=h.1vsw3ci) Requirements – Test Correspondence Summary 31

[21b](#_heading=h.2uxtw84) Acceptance Test Descriptions 31

[III](#_heading=h.1a346fx) Design 32

[22](#_heading=h.3u2rp3q) Design Goals 32

[23](#_heading=h.2981zbj) Current System Design 32

[24](#_heading=h.odc9jc) Proposed System Design 32

[24a](#_heading=h.38czs75) Initial System Analysis and Class Identification 32

[24b](#_heading=h.1nia2ey) Dynamic Modelling of Use-Cases 32

[24c](#_heading=h.2mn7vak) Proposed System Architecture 32

[24d](#_heading=h.11si5id) Initial Subsystem Decomposition 33

[25](#_heading=h.3ls5o66) Additional Design Considerations 33

[25a](#_heading=h.20xfydz) Hardware / Software Mapping 33

[25b](#_heading=h.4kx3h1s) Persistent Data Management 33

[25c](#_heading=h.302dr9l) Access Control and Security 33

[25d](#_heading=h.1f7o1he) Global Software Control 33

[25e](#_heading=h.3z7bk57) Boundary Conditions 34

[25f](#_heading=h.2eclud0) User Interface 34

[25g](#_heading=h.thw4kt) Application of Design Patterns 34

[26](#_heading=h.3dhjn8m) Final System Design 34

[27](#_heading=h.1smtxgf) Object Design 34

[27a](#_heading=h.4cmhg48) Packages 35

[27b](#_heading=h.2rrrqc1) Subsystem I 35

[27c](#_heading=h.16x20ju) Subsystem II 35

[27d](#_heading=h.3qwpj7n) etc. 35

[IV](#_heading=h.261ztfg) Project Issues 35

[28](#_heading=h.l7a3n9) Open Issues 35

[29](#_heading=h.1kc7wiv) Off-the-Shelf Solutions 35

[29a](#_heading=h.44bvf6o) Ready-Made Products 35

[29b](#_heading=h.2jh5peh) Reusable Components 35

[29c](#_heading=h.ymfzma) Products That Can Be Copied 36

[30](#_heading=h.3im3ia3) New Problems 36

[30a](#_heading=h.1xrdshw) Effects on the Current Environment 36

[30b](#_heading=h.4hr1b5p) Effects on the Installed Systems 36

[30c](#_heading=h.2wwbldi) Potential User Problems 36

[30d](#_heading=h.1c1lvlb) Limitations in the Anticipated Implementation Environment That May Inhibit the New Product 36

[30e](#_heading=h.3w19e94) Follow-Up Problems 36

[31](#_heading=h.2b6jogx) Migration to the New Product 37

[31a](#_heading=h.qbtyoq) Requirements for Migration to the New Product 37

[31b](#_heading=h.3abhhcj) Data That Has to Be Modified or Translated for the New System 37

[32](#_heading=h.1pgrrkc) Risks 37

[33](#_heading=h.49gfa85) Costs 37

[34](#_heading=h.2olpkfy) Waiting Room 37

[35](#_heading=h.13qzunr) Ideas for Solutions 37

[36](#_heading=h.3nqndbk) Project Retrospective 38

[V](#_heading=h.22vxnjd) Glossary 38

[VI](#_heading=h.i17xr6) References / Bibliography 38

[VII](#_heading=h.320vgez) Index 38

### **List of Figures**

[Figure 2 - Sample Use Case Diagram from Bruegge & DuToit ( modified ) 16](#_heading=h.19c6y18)

[Figure 3 - Sample Use Case Diagram from Robertson and Robertson 17](#_heading=h.3tbugp1)

### **List of Tables**

[*Table 2 - Requirements - Acceptance Tests Correspondence* 31](#_heading=h.4fsjm0b)

# Project Description

## Project Overview

The product of this project is a mobile device application that is an accurate GPS mapping of trails in National Parks and forest preserves. A user will be able to view all of the trails in a given park and be able to select what trails they want to go on. They will also be able to see where all the landmarks are within the park and design a route that can bring them to those landmarks. We also want the users to help improve the parks and preserves, so the app will include a rating and commenting system for parks. Having this feedback will help parks focus on the things that are done well and things that are done poorly. The GPS system will be built using Google’s GPS API that they use for their Maps application.

## The Purpose of the Project

The purpose of this project is to improve the National Park system. Millions of people visit these parks every year and we want to make their experience the best it can be. By making it more accessible for all people, we can spur the growth of the National Parks community.

### The User Business or Background of the Project Effort

The business that the application is being built for is the National Park Service. The people that work these parks certainly are familiar with all the routes in the park, but most visitors are not familiar at all. This application will give the users the tools to navigate just like a park ranger. The motivation for this project is to have more people come visit these places in an easier than ever way. Having more people wanting to travel to these places will generate more money for the parks and for the local businesses. Navigating unfamiliar places can be daunting for many people which is why having an easy-to-use tool like this would be important to create.

### Goals of the Project

We want to improve the experience of traveling to and exploring a National park for our clients.

### Measurement

The way we can measure the success of the project would be through the total number of visitors to National parks. By making the process easier for people to plan we should be able to directly see the increase in the number of people that come to the parks.

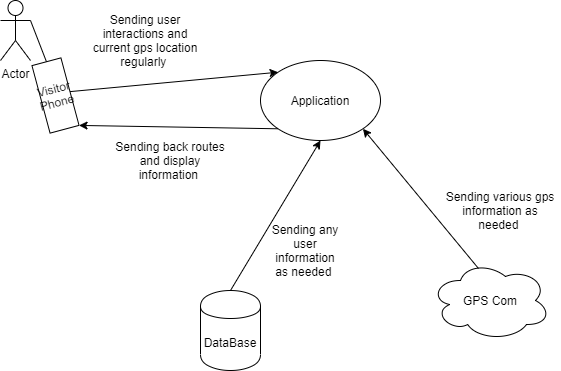
## The Scope of the Work

The work addressed by this project will be for the entertainment and guidance of national park visitors. It would be a helpful tool that would improve the experience of the national park.

### The Current Situation

Park visitors are now wasting time figuring out a route to best suit the areas they would want to visit at the park. Instead of using a paper map to figure it out, an application that could quickly be used will allow for more time to enjoy the park.

### The Context of the Work



The actor will be any person such as a park visitor, park employees, or management. The management will also have the ability to send data that may change the database.

### Work Partitioning

|  |  |  |
| --- | --- | --- |
| **Event Name** | **Input and Output** | **Summary** |
| 1. Reading visitor/ employee/ management location | Application(in)  Gps Com(out) | Get current user location at first and continually update location as needed |
| 1. Management edit available locations | Database(in) | The database will edit the currently available locations |
| 1. Application display list of locations | Application(out), Database(out) | Application sending the user the currently available locations |
| 1. Planning new areas | Database(in) | Sending new information to database about new locations |
| 1. Private Areas | Application(out), Gps (out) | Informing the users that they are in a private location |

### Competing Products

There are currently no competing products with ours. The reason for this is because there are no applications that use the current gps mapping to edit the current area of a national park.

## The Scope of the Product

The application will be used to manage and guide through national parks. Only the administrator will be able to change the available locations and create new paths. The employees and visitors will have access to looking up paths and locations that are open to the public. While the visitors will have access to

### Scenario Diagram(s)

### 

### Product Scenario List

* Find Best Route
* Edit Path
* Edit prohibited areas
* Recommend new routes

### Individual Product Scenarios

Find Best Route- The user will select the option “Find Best Route”. In this option the user will select the areas of the park that they would like to visit. The application will then produce the most optimal route based on the areas picked. Then that route will be shown to the user and the user will have the option to edit the list if they wanted.

Edit Path- Only the key user profiles will have the option to edit a certain path. In this option they will have the ability to edit the path of their choice in a way that would see fit. After the user is done editing the path it can then be uploaded to the main database to be changed.

Edit Prohibited Paths- Only the key user profiles will have the option to change the current prohibited pathways for basic users. The application will look similar to the edit path option, only with a new option to change a current available path to change it to prohibited. After that is done in the database that path will change status to the off limits.

Recommend new Paths- All users will be able to use this feature. The feature will allow the user to recommend paths for others to follow at the park. When the user selects the option they will be able to add an description of why they would recommend that path. They will then choose to publish that path to the main database to be shared with others.

## Stakeholders

### The Client

The client is the National Park Service. They will pay for the development of the GPS trail application and provide input and guidance for the product.

### The Customer

The end product will be distributed through the app store on phones. It is not intended that the customer will purchase the app, as the goal is to have as many people as possible hiking the trails.

### Hands-On Users of the Product

The hands-on users will be those who download the app. They will be visitors in the park who wish to hike the trails. These users are expected to know the park’s features and attractions, but they are not necessarily expected to be familiar with the park’s geography. It is expected for the hands-on users to be at least 16 years of age.

### Maintenance Users and Service Technicians

Maintenance will be necessary to ensure that the pathing algorithm is working properly and the trail map is up to date. Trails in the park may closed or new paths may open. This maintenance is expected to be done by the National Parks Service.

### Other Stakeholders

There will be legal stakeholders. They will be interested in making sure the app complies with any relevant laws.

### User Participation

Park workers will provide feedback during the development of the product. They will ensure that the generated trails are accurate and safe for the end-users of the product.

### Priorities Assigned to Users

Park workers are the key users of the product. They have detailed knowledge of the park and can ensure the app is working correctly. The park visitors are secondary users. They will be the users the app is ultimately targeted towards.

## Mandated Constraints

### Solution Constraints

The program must be programmed to run on a mobile device. Creating the product to run on a desktop is not ideal because it’s intended to be marketed towards hikers, joggers, and other clients who visit national parks. Creating a mobile application is the only viable option for a product that expects the users to be on the move.

### Implementation Environment of the Current System

Because the product is going to be used in national parks, expecting the client to have a stable internet connection while in the park is asking too much. For this reason, the app will need to be made as native as possible. Designing the application so that it’s as native as possible alleviates the reliance on a stable internet connection.

The maps of each national park will be downloaded during installation of the product. One of the few instances where the application will require an internet connection is when it’s attempting to figure out which national park the user is in. Drawing the path the user will take will also require an internet connection.

### Partner or Collaborative Applications

The application will primarily rely on Google’s API in order to display the possible paths the user can take when the application is given a starting point. The maps for each national park will be stored as a .shp in order to ensure compatibility with Google maps.

Other collaborative applications include the users “phone” app, which can be used to call for help in the event of an emergency with a simple press of a button in the application.

### Off-the-Shelf Software

Google’s API and SDK allows the application to run by loading .shp and showing the user the entire map of the national park they are currently in. Google’s API must be used in order for the application to run since its primary purpose is to show the user a possible path that they can take. The user is also expected to be running IOS or Android, as these are the only two operating systems that the application will support.

### Anticipated Workplace Environment

Since this application is primarily marketed towards those who visit national parks, the environment of a national park does not allow for stable internet connection. For this reason, a primary goal during development should be to limit the reliance on an internet connection and to make things as native as possible. Having everything pre-downloaded from the beginning will ensure that the application remains working even when the user has no internet connection.

### Schedule Constraints

The most popular times for national parks to receive visitors is during the summertime. For this reason, the beta version of the application should be finished by the beginning of April and the second version of the application should be finished by mid May.

The main features that should be implemented from get-go is the user interface and the ability to locate which national park the user is currently in. These are the two most basic things that should be added. From there, the ability for the application to read .shp files and load them depending on which national park the user is located in should be the next thing to be added. The final feature to be added is presenting the paths to the user based on the user's current location with the help of the phones location services and Google’s API. The app with these basic features is considered in beta.

The second iteration of the application will allow the user to select their own starting point rather than relying on their location services in the event that they do not have an internet connection. Generating a path will also be possible natively when the user manually chooses a starting point and an ending point.

### Budget Constraints

An application like this does not require a major budget. The biggest expense to keep the application running is the database that holds the national parks information. Paying Google to make API calls will also be a primary expense since the application will constantly be making calls in order to show the user's current location and their progress as they navigate through the path.

## Naming Conventions and Definitions

### Definitions of Key Terms

Actor: Any person utilizing the app.

Key users: Users critical to the continued success of the product

Secondary users: Users use the product, but their opinion is not critical to the product long-term

### UML and Other Notation Used in This Document

This document follows the standard UML file format.

### Data Dictionary for Any Included Models

There is no data dictionary for any of the models.

## Relevant Facts and Assumptions

### Facts

The time when national parks receive the most visitors is during the summer so the application should be completed by that time. Most national parks can also have multiple paths that the user will be able to select from. The application will use Google’s API in order to generate a path for the user. The maps of the national parks will also be downloaded during the installation of the application. The users device running the application will be either an iPhone running IOS or an Android running any android operating system that allows .apk files to be installed.

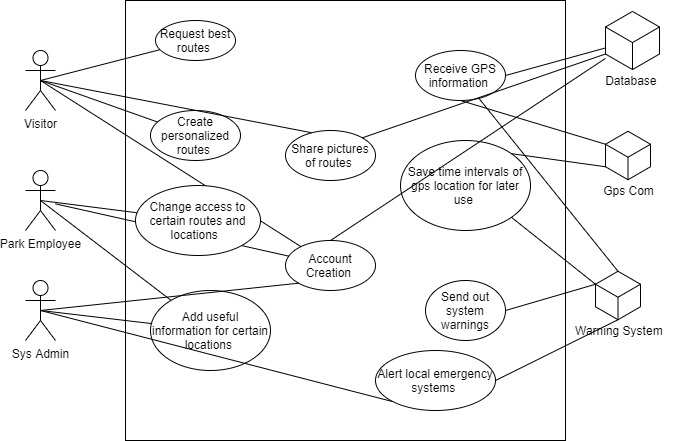
### Assumptions

It’s assumed that the user is not going to have a solid internet connection so the app will run as natively as possible to avoid the limitations of having no internet connection. It is also assumed that while the internet is not the best, it will still be good enough for Google API to run and to show the user their current location from the beginning of the path to the end.

# Requirements

## Product Use Cases

### Use Case Diagrams



### Product Use Case List

* Request best routes
* Warning local emergency systems
* Account creation
* Photo sharing
* Photo posting
* Change location access
* Save interval time gps location
* Send out warning to all users
* Edit relevant information about route
* Comments about routes
* Creation of personal routes
* Voting systems of best routes
* Different routes on same trail
* Animal sightings
* Animal information

### Individual Product Use Cases

|  |
| --- |
| Use case ID: 134 Name: Animal Sightings  pre-conditions: Gps Communication is up, Database connection is fine, Warning Systems online  post-conditions: Gps Communication is updating the database, database updates, warning system sends out alerts  Initiated by: Admin or Employee Users  Triggering Event: Learned of dangerous animal sightings  Additional Actors: |
| Sequence of Events:  The actor should click on the “Animal Emergency” option in the main application.  The system then displays a menu, asking for relative information about the animal  The actor will then select and type relative information about the animal and submit afterwards.  The system will then save that information and display a screen showing the local surround warning systems available to alert.  The actor will then select which warning options they will like to alert and submit afterwards.  The system will then alert the warning systems and display to the user the process of those alerts.  . |
| Alternatives: An sound system being activated to warn.  Exceptions: The system communication system or warning system is not on. |

## Functional Requirements

**2.1 - Trail Map**

**Description:** The program displays a map of the trail that is generated.

**Rationale:** This is the main functionality of the program. Users must be able to see and understand the trail on a map of the park so that they can follow it on their hike.

**Fit Criterion:** A trail map is created which is overlaid on top of an overall map of the park.

**Acceptance Tests:** Test 11

## Data Requirements

**3.1 - Account Creation**

**Description:** The system must allow the user to create an account in the app

**Rationale:** The user wants to be able to store their saved data to a specific account such as favorite routes, pictures uploaded, and parks visited.

**Fit Criterion:** The user will be able to login to their account successfully.

**Acceptance Tests:** Test 12

**3.2 - Registration form**

**Description:** The system must allow the user to input their first name, last name, email, birthday, and phone number into the registration form to create an account. Email must be in the form ‘username@domain’ and birthday must be in the form month-day-year. Phone number must only be 10 digits.

**Rationale:** The user wants to have their own data tied to a specific account

**Fit Criterion:** The user will be able to input their data into the registration form.

**Acceptance Tests:** Test 12

**3.3 - Popular Route**

**Description:** The system must allow the user to select the best route based on the data from the route votes.

**Rationale:** The user wants to go through the most popular routes.

**Fit Criterion:** The user will be able to walk through the most popular route that was voted by other users.

**Acceptance Tests:** Test 13

**3.4 - Route Comments**

**Description:** The system must give the user the option to view comments about a route.

**Rationale:** The user wants to read what other people thought of a specific route.

**Fit Criterion:** The user will be able to read the comments that other users have left about a route.

**Acceptance Tests:** Test 14

## Performance Requirements

### Speed and Latency Requirements

**4.1 - Loading Registration Form**

**Description:** The system must load the registration form within 5 seconds and present it to the user (assuming the user has a decent internet connection).

**Rationale:** Users don’t want to wait a long time just to register an account.

**Fit Criterion:** The user is able to start inputting their data into the registration form within 5 seconds of the form being loaded.

**Acceptance Tests:** Test 16

**4.2 - Current User Location**

**Description:** The system must periodically update the users current location while going through a route in 10 seconds or less

**Rationale:** The user wants to see their current progress through the route and their current location in the route so they know where to go

**Fit Criterion:** The user is able to see their location change within the app as they go through a route.

**Acceptance Tests:** Test 17

### Precision or Accuracy Requirements

**4.3 - User Location**

**Description:** The system must pinpoint the user's location with a 10 foot room for error.

**Rationale:** The user wants to see an accurate location of where they currently are.

**Fit Criterion:** The user is able to see where they currently are in the forest and where to go.

**Acceptance Tests:** Test 18

### Capacity Requirements

**4.4 - Number of Routes**

**Description:** The system must only load one route maximum at a time per device. If the user wishes to load a new route, they must cancel the current navigation for the route.

**Rationale:** The user doesn’t want to walk through multiple routes at the same time.

**Fit Criterion:** The user will be able to load a single route at a time.

**Acceptance Tests:** Test 21

**4.5 - Number of Pictures**

**Description:** The system must only load 5 pictures at a time for a specific park. If they wish to load more images, they must click the “load more” button

**Rationale:** The user doesn’t want to waste internet data loading images they may not want to view. The user wants to be able to choose whether or not to load more images.

**Fit Criterion:** The user will be able to view 5 pictures of the park and given the option to load 5 more images.

**Acceptance Tests:** Test 22

## Dependability Requirements

### Reliability Requirements

**ID 5.1 - RunOffline**

**Description:** The system needs to have reliable steps to save important information if the system goes down.

**Rationale:** The system needs steps to recover saved gps and database information.

**Fit Criterion:** This is required to make sure the system is able to update itself after it is able to establish communications again. Also, to show the warning systems that a restart is needed to the user.

**Acceptance Tests: Test 1**

### Availability Requirements

**ID 5.2 - GpsInformation**

**Description:** A check to make sure that the system is saving gps information actively to the database.

**Rationale:** At all time the system should be saving information actively just in case of a shutdown.

**Fit Criterion:** When checking the system while offline the user should be able to see certain functions with the last information collected before it went offline.

**Acceptance Tests: Test 1**

### Robustness or Fault-Tolerance Requirements

**ID 5.3 - Offline Access**

**Description:** The system should always still have important information to run certain functions.

**Rationale:** Just in case that the system goes offline the systems will still be able to display and access some previous data.

**Fit Criterion:** Information such as gps locations and the system’s warning information should be able to be accessed by the admins of the system.

**Acceptance Tests:** Test 2

### Safety-Critical Requirements

**ID 5.4 - ProperityCode**

**Description:** The system is actively checking all users locations.

**Rationale:** Need to know all persons on the property location to make sure that location’s rules are being followed

**Fit Criterion:** The system will need to run tests throughout to make sure that certain rules of certain locations are being followed. Should have a way to show that information easily to an admin user.

**Acceptance Tests:** Test 2

## Maintainability and Supportability Requirements

### Maintenance Requirements

**6.1 - Maintainability**

**Description:** The system needs to be able to be accessed by a server admin who will perform any necessary maintenance to the system.

**Rationale:** There needs to be a way to restart the system, apply small patches, and update the version.

**Fit Criterion:** The system has a means to be shut down and turned on

**Acceptance Tests:** Test 33

### Supportability Requirements

**6.2 - Product Support**

**Description:** Secondary users will be provided support for potential problems they meet during use. Support may be through online or in person at national parks.

**Rationale:** We want the best experience possible for our users. Providing support for problems is one way to do so.

**Fit Criterion:** Secondary users are able to contact a support personnel through online and/or in person at a national park.

**Acceptance Tests:** Test 28

### Adaptability Requirements

**6.3 - Adapting**

**Description:** The system should be able to be adapted to iOS and Android operating systems

**Rationale:** We want the most amount of visitors the opportunity to use our product.

**Fit Criterion:** The product is usable on both android and iphones.

**Acceptance Tests:** Test 23

### Scalability or Extensibility Requirements

**6.4 - Operation Scaling**

**Description:** The system should support an increase in use from its first year to the second.

**Rationale:** As our usage base grows, the system must still be able to support all requests in a timely manner.

**Fit Criterion:** The system maintains a constant response time after an 30% increase in requests.

**Acceptance Tests:** Test 24

### Longevity Requirements

**6.5 - Life Expectancy**

**Description:** Specifies how long the product is expected to be in use.

**Rationale:** The product must work correctly for its whole life cycle.

**Fit Criterion:** The system continues to be usable after five years.

**Acceptance Tests:** Test 25

## Security Requirements

### Access Requirements

**7.1 - Admin Access**

**Description:** Specifies what data system maintenance administrators have in the system.

**Rationale:** System maintenance administrators need park data to maintain the system correctly. However, our clients’ privacy must still be protected.

**Fit Criterion:** When the system is accessed by an administrator account, the system allows them to view all park data in the system. They can see aggregate client data, but not information pertaining to a specific customer.

**Acceptance Tests:** Test 7

### Integrity Requirements

**7.2 - Ensuring Park Information**

**Description:** Test that the system data does not become corrupted.

**Rationale:** Avoiding data corruption is essential to ensure correct information is given to the clients.

**Fit Criterion:** System data does not change when it is not asked to change. Data must be consistent between sessions.

**Acceptance Tests:** Test 26

### Privacy Requirements

**7.3 - User Privacy**

**Description:** Makes sure users do not have access to data pertaining to any other user.

**Rationale:** User privacy is critical. Secondary users should not have access to user data other than their own.

**Fit Criterion:** Secondary users can not see information that is not related to themselves.

**Acceptance Tests:** Test 7

### Audit Requirements - Does not apply

### Immunity Requirements

**7.5 - Virus immunity**

**Description:** The system must be resistant to computer viruses.

**Rationale:** The system must be secure and not vulnerable to malicious software.

**Fit Criterion:** Your fit criteria here . . .

**Acceptance Tests:** Test 27

## Usability and Humanity Requirements

### Ease of Use Requirements

**8.1 - Account Creation Time**

**Description:** The system must allow the user to create a new account within 3 minutes of loading the registration form.

**Rationale:** Users want to be able to easily create a new account.

**Fit Criterion:** The user will have a working account within 3 minutes of opening the registration form.

**Acceptance Tests:** Test 29

**8.2 - Route Selection**

**Description:** The system must be able to show all the available routes as soon as the app detects which park the user is in.

**Rationale:** Users want to select a route as quickly as possible.

**Fit Criterion:** The user will be able to select which route to go through within 10 seconds of entering a park.

**Acceptance Tests:** Test 30

### Personalization and Internationalization Requirements

**ID 8.3 - Profile Customization**

**Description:** The system must allow a user to customize their profile

**Rationale:** The main idea of the app is to have a social media like experience, so being able to personalize your profile is a must

**Fit Criterion:** There is a profile editing feature where the user can change their username, description, edit their routes of a park , and add pictures of those said routes

**Acceptance Tests:** Test 3

### Learning Requirements

**8.4 - Forest Information**

**Description:** The user must be able to comfortably load information about the park such as images and comments that other users have made.

**Rationale:** The user wants to be able to load information about a park effortlessly.

**Fit Criterion:** The user is able to quickly load information about the park within 3 minutes of selecting a park.

**Acceptance Tests:** Test 31

### Understandability and Politeness Requirements

**8.5 - Park Selection**

**Description:** The system must allow the user to select a national park and load information about the park and the option to load routes.

**Rationale:** The user wants to be instructed on how to use the app one step at a time. The first step would be selecting a national park.

**Fit Criterion:** The user is able to select a park and view information about the park upon opening the app.

**Acceptance Tests:** Test 5

### Accessibility Requirements

**8.6 - Colorblind Support**

**Description:** The system must allow the user to change to colorblind mode for that app to change the text color and the GPS screen color

**Rationale:** A colorblind user doesn’t want to have trouble using the app and reading the screen

**Fit Criterion:** The colorblind user will be able to more comfortably read the screen

**Acceptance Tests:** Test 20

### User Documentation Requirements

**8.7 - FAQ**

**Description:** The system should allow the user to view the FAQ in the settings.

**Rationale:** The user wants their questions answered should any arise during the use of the app.

**Fit Criterion:** The user will be able to read frequent questions and answers and have their question answered.

**Acceptance Tests:** Test 19

### Training Requirements

**8.8 - Tutorial**

**Description:** The system should load a tutorial for the user if this is their first time using the app.

**Rationale:** The user wants to be shown how to use the app most efficiently.

**Fit Criterion:** The user will have a good feel for how to use the app and be able to perform basic functions such as selecting a national park and loading a route.

**Acceptance Tests:** Test 32

## Look and Feel Requirements

### Appearance Requirements

**ID 9.1 - Style Change**

**Description:** The system should be able to adjust to the user's preferences of system appearance.

**Rationale:** Every user has a certain preference of colors and fonts that should be added to add to a useful experience.

**Fit Criterion:** The system will have a simple process to change certain user style preferences in an option menu style.

**Acceptance Tests:** Test 4

### Style Requirements

**ID 9.2 - EaseOfUse**

**Description:** The system should display primary use functionalities easily to the user

**Rationale:** Users should not have to go through confusing menus to use certain functions. Want to make sure that the users have a seamless experience of using the application.

**Fit Criterion:** That the user should be able to quickly find and switch between the main system’s functionalities. Other functions should be visible from another in order to easily switch between them.

**Acceptance Tests:** Test 4

## Operational and Environmental Requirements

### Expected Physical Environment

**ID 10.1 - Connectivity**

**Description:** The system needs to handle a user losing internet connection while using the app

**Rationale:** Some national parks are in mountains or in thick-wooded forests, so there will be a loss of internet connection at times.

**Fit Criterion:** The system will locally store data on the user’s device and once it regains internet connection, the data will be sent to the system to update.

**Acceptance Tests:** Test 2

### Requirements for Interfacing with Adjacent Systems

**ID 10.2 - Google API**

**Description:** The system needs to interface with Google API

**Rationale:** This app uses Google’s GPS system, so using their API is mandatory

**Fit Criterion:** Having access to Google API function calls

**Acceptance Tests:** Test 10

### Productization Requirements

**ID 10.3 - Where to Download**

**Description:** The system is to be distributed on the App Store and Google Play Store

**Rationale:** Users need to be able to download the application to their respected devices

**Fit Criterion:** The app arises in a search for it in each respected app store.

**Acceptance Tests: Test** 15

### Release Requirements

**ID 10.4 - Version Control**

**Description:** The system must be up to date with the current version of iOS and Android

**Rationale:** These operating systems update frequently, so our app must be compatible with current versions so it can be used

**Fit Criterion:** A number will be kept that corresponds to the operating system version it is running on.

**Acceptance Tests:** Test 8

## Cultural and Political Requirements

### Cultural Requirements

**ID 11.1 - Language of Spelling**

**Description:** The system will use the U.S. English form for all spelling.

**Rationale:** The application is designed for the National Park Service which is only in the United States, so the language should be in U.S. English

**Fit Criterion:** Any text that is not a user’s name will be in U.S. English

**Acceptance Tests:** Test 9

### Political Requirements

**ID 11.2 - Powered by Google**

**Description:** The system needs to give credit to Google for using their maps API

**Rationale:** The main functionality of the application relies solely on Google API, so they would most certainly need to have some form of recognition.

**Fit Criterion:** In the FAQ there is an explanation of how Google is involved in the software

**Acceptance Tests:** Test 19

## Legal Requirements

### Compliance Requirements

**ID# 12.1 - Information Privacy**

**Description:** The system must comply with the California Consumer Privacy Act

**Rationale:** In the state of California users must be informed about what data is being collected and shared, and give them a right to access, to delete, and to opt-out

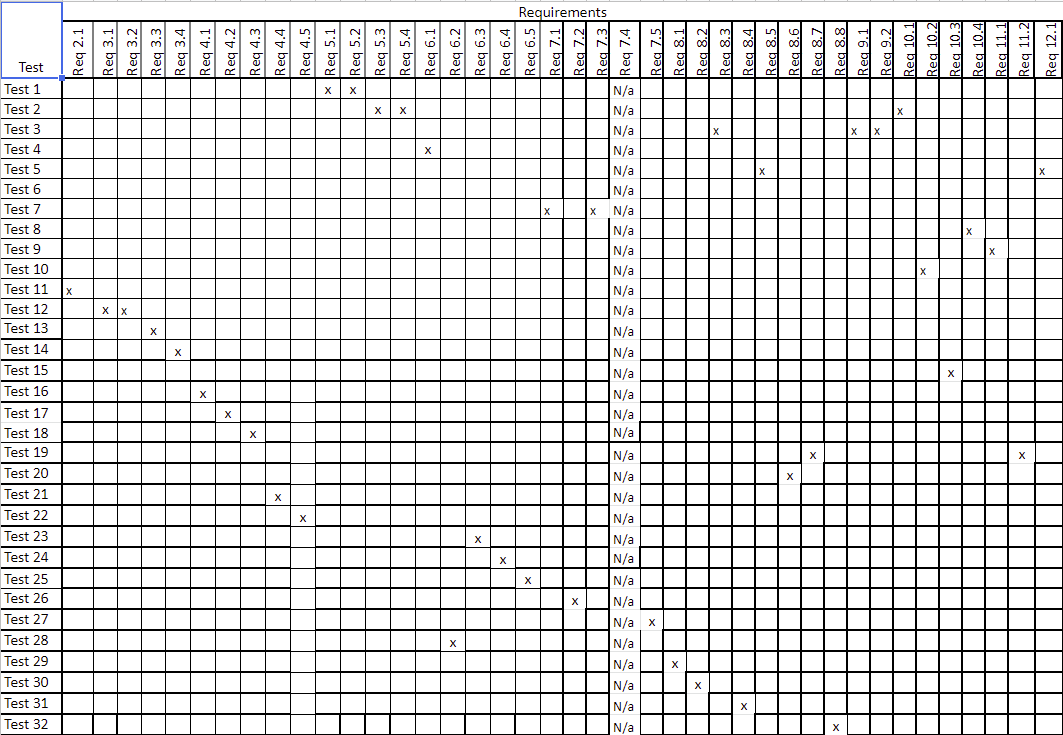
**Fit Criterion:** When creating an account, there will be a disclaimer about the information that will be collected about the user and they can choose to not have their data not collected

**Acceptance Tests:** Test 6

### Standards Requirements - Does not apply

## Requirements Acceptance Tests

### Requirements – Test Correspondence Summary

**

### Acceptance Test Descriptions

Test 1: This test will make sure that the system is saving important information actively as the system is online. 5.1, 5.2

Test 2: The test will make sure that when the system is offline that it will have access to the required functions and the data collected is stored locally. 5.3, 5.4, 10.1

Test 3: Test will make sure that the system changes appearance according to the current users settings. And that the application user face shows the primary functions properly. 9.1, 9.2, 8.3

Test 4: This test turns a test server on and off where it makes changes when turned off and we see those changes when turned back on 6.1

Test 5: Given a few different parks, it checks to make sure that the correct information is loaded upon selected a specific park

Test 6: This test processes a group of fake users who either have their data collected or not. It makes sure that only the people who want their data collected have their statistics added to a total. 12.1

Test 7: The test will make sure that user privacy is protected. Information concerning a user will not be available to users that it should not be. 7.1,7.3

Test 8: This test will do a check on the version number of the app and the current OS version. 10.4

Test 9: Runs a check on some text to ensure it is in the U.S. English form. 11.1

Test 10: This test calls the Google API functions that will be used in the app and makes sure that they work as expected. 10.2

Test 11: This test will check to see if a path has been generated. 2.1

Test 12: This test will check if the user is able to create an account in the registration form. 3.1, 3.2

Test 13: This test will check if the most popular trail is shown to the user based on popularity. The most voted for trail should be the one generated. 3.3

Test 14: This test will check to see if the comments on a route are successfully shown to the user. 3.4

Test 15: This test checks the App Store and Google Play Store for the app. 10.3

Test 16: This test will check to see if the registration form is loaded within 5 seconds. 4.1

Test 17: This test will check to see if the users location is periodically retrieved within 10 seconds while going through a route. 4.2

Test 18: This test will check to see if the system is able to pinpoint the users location within a 10 foot room for error. 4.3

Test 19: Tests the functions that display the text on the FAQ page and that they print the correct things. 8.7 11.2

Test 20: Tests color changing on a few different images to make sure it changes to the correct colorblind type (Deuteranomaly, Protanomaly, Protanopia). 8.6

Test 21: This test will check to see if the system prevents a user from loading 2 or more routes at a time. 4.4

Test 22: This test will check to see if the system only loads 5 images at a time for a park and if the user is able to load 5 more images at a time. 4.5

Test 23: This test will check to see if the IOS and Android versions of the app show the same data and behave the same way. 6.3

Test 24: This test will ensure that the system is able to handle an influx of users using the app at once. 6.4

Test 25: This test will ensure that the system continues to function after each development cycle. Must maintain a life of 5 years. 6.5

Test 26: This test will ensure the data used is still able to be used is still correct. 7.2

Test 27: This test ensures that there has been no data breach or other viruses on the app. 7.5

Test 28: Tests to make sure that the system can open the support menu and be able to send a support ticket with their issue 6.2

Test 29: This test ensures the user is able to make an account on the app within 3 minutes of the registration form loading. 8.1

Test 30: This test ensures all the routes are presented to the user upon entering the park. 8.2

Test 31: This test ensures that the app is easy to use by showing all the UI elements in an easy to follow format and the user is able to understand what to do upon opening the app. 8.4

Test 32: This test ensures that the user is presented with a tutorial the first time they open the app. 8.8

# Design

## Design Goals

*SV: Identify the important design goals that are to be optimized in the proposed design.*

Your text goes here . . .

## Current System Design

*SV:* ***IF*** *the proposed new system is to replace an existing system, then the current system should be described here. Otherwise insert a brief statement that there is no pre-existing system.*

Your text goes here . . .

## Proposed System Design

*This section will make heavy use of class diagrams, and also sequence and deployment diagrams where noted. However don’t overlook finite state, activity, communication, or other diagram types as needed for effective communication.*

### Initial System Analysis and Class Identification

*SV: Perform grammatical and similar analyses to identify the most import and obviously needed classes, and to organize them into an initial class structure. An initial class diagram is appropriate, containing few if any internal details.*

Your text goes here . . .

### Dynamic Modelling of Use-Cases

*SV: Insert sequence diagrams of ( at least the most important ) use-cases, as a means of identifying other needed classes.*

Your text goes here . . .

### Proposed System Architecture

*SV: Identify the Software Architecture to be applied to this project, such as Client-Server, Repository, MVC, etc., along with justification for the choice.*

Your text goes here . . .

### Initial Subsystem Decomposition

*SV: A slightly more detailed class diagram, showing the classes identified in sections 24a, 24b, and 0 above, partitioned into subsystems. For each subsystem provide a brief description of the subsystem, including its key responsibilities. There should still be few if any internal details.*

Your text goes here . . .

## Additional Design Considerations

*SV: The sections listed here do not need to be presented in the order given, and may not all be relevant for any particular project. Those that are relevant can help identify additional classes that are needed as a result.*

### Hardware / Software Mapping

*SV: This is particularly important for distributed systems, such as those employing a client-server architecture. Use a deployment diagram to indicate which subsystems are mapped onto which piece(s) of hardware, and what communication subsystems need to be added to the system as a result.*

Your text goes here . . .

### Persistent Data Management

*SV: Document the classes and perhaps subsystems necessary to store persistent data when the system shuts down, and to restore that data when the system starts back up again.*

*Reiterate key data structures and information as necessary for the understanding of this design phase. Refer the reader back to the data dictionary in section* ***Error! Reference source not found.*** *to avoid undue repetition, while reviewing only the most relevant items here.*

Your text goes here . . .

### Access Control and Security

*SV: Identify the access control and security concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns.*

Your text goes here . . .

### Global Software Control

*SV: Identify the global software control concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns.*

Your text goes here . . .

### Boundary Conditions

*SV: Identify the boundary condition concerns for this system, and the new classes and/or subsystems that must be added to handle those concerns. In particular consider startup, shutdown ( normal or abnormal ), and the creation and/or maintenance of any configuration files, databases, or similar supporting data files.*

Your text goes here . . .

### User Interface

*SV: Include a preliminary user interface design here, possibly as a rough sketch or other mockup, in order to identify additional classes needed to implement the interface.*

Your text goes here . . .

### Application of Design Patterns

*SV: Any design patterns applied as a result of previous sections should have been addressed there, and identified as such at the time. Use this section to document only the additional design patterns that were not previously covered elsewhere. ( If any. )*

Your text goes here . . .

## Final System Design

*SV: Include here the final version of the overall system design, incorporating all the subsystems and classes added as a result of additional design considerations. Multiple diagrams may be needed, possibly starting with an overall package diagram showing all the different subsystems and the ( important ) classes contained within each one. Still not a lot of internal details.*

Your text goes here . . .

## Object Design

*This section documents the internal details of each class, to the extent that they can be designed at this time. Included should be the class interfaces ( public method signatures and responsibilities ) and constraints. It is probably best to break this section up into subsections corresponding to subsystems as documented above, and/or by ( Java ) packages if those are designed. It may also be appropriate to address additional design pattern considerations here, but not to the point of being redundant of previous documentation.*

*Certain methods, such as simple getters, setters, and constructors are not always documented, unless there is something special about them such as in the Singleton or Factory Method design patterns.*

### Packages

*SV: If the design involves assigning classes to packages ( .e.g Java packages ), then the packages to be created should be documented here.*

Your text goes here . . .

### Subsystem I

Your text goes here . . .

### Subsystem II

Your text goes here . . .

### etc.

Your text goes here . . .

# Project Issues

## Open Issues

*SV: Issues that have been raised and do not yet have a conclusion.*

Your text goes here . . .

## Off-the-Shelf Solutions

*SV: Discussion of products or components currently available that could either be incorporated into the new solution or simply used instead of developing ( parts of ) the new solution.  The distinction between sections 35 a, b, and c is subtle, and not very important.*

Your text goes here . . .

### Ready-Made Products

*SV: Products available for purchase that could be used either as part of a solution or instead of ( a part of ) a solution.*

Your text goes here . . .

### Reusable Components

*SV: Similar to 35a, but for components such as libraries or toolkits instead of fully blown products.*

Your text goes here . . .

### Products That Can Be Copied

*SV: Products that could legally be copied would typically be past projects developed by the same development group, provided there were no restrictions that would prevent their reuse.*

Your text goes here . . .

## New Problems

*SV: The proposed new system certainly has its benefits, but it could also raise new problems.  It is a good idea to identify any such potential problems early on, rather than being surprised by them later.*

### Effects on the Current Environment

*SV: Could the new system have any adverse effects on the working environment, e.g. the way people do their jobs?*

Your text goes here . . .

### Effects on the Installed Systems

*SV: Could the new system have any adverse effects on other hardware or software systems?*

Your text goes here . . .

### Potential User Problems

*SV: Could the new system have any adverse effects on the users of the software? Could users possibly have a negative response to the new system?*

Your text goes here . . .

### Limitations in the Anticipated Implementation Environment That May Inhibit the New Product

*SV: Are there any ( physical ) limitations in the expected environment that could inhibit the proposed product?  ( e.g. weather, electrical interference, radiation, lack of reliable power, etc. )*

Your text goes here . . .

### Follow-Up Problems

*SV: Basically any other possible problems that could occur.*

Your text goes here . . .

## Migration to the New Product

*SV: This section only applies when there is an existing system that is being replaced by a new system, particularly when data must be preserved and possibly translated / reformatted.  Otherwise just write "Not Applicable" under section 38 and remove sections 38a and 38b.*

### Requirements for Migration to the New Product

*SV: These are a list of requirements relevant to the migration procedures.  For example a requirement that the two systems be run in parallel for a time until the client is satisfied with the new system and the users know how to use it.*

Your text goes here . . .

### Data That Has to Be Modified or Translated for the New System

*SV: This section specifically addresses****data****that must be preserved and/or translated / reformatted during the migration process.*

Your text goes here . . .

## Risks

*SV: Consideration of the potential risks that could cause the project to fail / underperform.*

Your text goes here . . .

## Costs

*SV: An estimate of what it will cost to complete this project.  Think not only in terms of dollars, but also time, resources, lost opportunities, etc.*

Your text goes here . . .

## Waiting Room

*SV: This is a place to record ideas or wishes that will not be included in the current release of the product, but which might be worth reconsidering at a later date.*

Your text goes here . . .

## Ideas for Solutions

*SV: When developing requirements only, it is not the role of the business analyst to dictate the implementation of the solution.  However they can pass along any ideas they have here as suggestions to the developers.  For CS 440 this report includes system and object design, so this section would make suggestions for implementation and testing that would come after design, such as the use of a particular language, IDE, library, or other tools.*

Your text goes here . . .

## Project Retrospective

*SV: At the conclusion of the ( CS 440 ) project, reflect back on what worked well and what didn't, and how the process could be improved in the future.*

Your text goes here . . .

# Glossary

*SV: The glossary is a more complete and inclusive dictionary of defined terms than that found in section I.7.a, the latter of which only covered the most important key terms needed to understand the report.*

Your text goes here . . .

# References / Bibliography

*This section describes the documents and other sources from which information was gathered. This sample bibliography was generated using the “Insert Citation” and “Bibliography” buttons in the “Citations & Bibliography” section under the “References” tab of MS Word. Creating new citations will not update this list unless you click on it and select “Update Field”. You may need to reset the style for this paragraph to “normal” after updating.*

|  |  |
| --- | --- |
| [1] | Robertson and Robertson, Mastering the Requirements Process. |
| [2] | A. Silberschatz, P. B. Galvin and G. Gagne, Operating System Concepts, Ninth ed., Wiley, 2013. |
| [3] | J. Bell, "Underwater Archaeological Survey Report Template: A Sample Document for Generating Consistent Professional Reports," Underwater Archaeological Society of Chicago, Chicago, 2012. |
| [4] | M. Fowler, UML Distilled, Third Edition, Boston: Pearson Education, 2004. |

# Index

*This section provides an index to the report. The sample below was generated using the “Mark Entry” and “Insert Index” items from the “Index” section on the “References” tab, and can be automatically updated by right clicking on the table below and selecting “Update Field”. To remove marked entries from the document, toggle the display of hidden paragraph marks ( the paragraph button on the “Home” tab ), and remove the tags shown with XE in { curly braces. }*

Design 61, 63

Requirements 35, 51, 58

Test 64, 65