Lab 2 – Load.In Product Specification

Chris Miller

Old Dominion University

CS 411

Professor Janet Brunelle

28 March 2021

Version 1

Table of Contents

Tables and Figures

1. Introduction

The average American moves 11.7 times in their lifetime. In 2019 nearly 31 million Americans moved, which was 9.8 percent of all Americans. The average cost of an interstate move is $4,100 and an in-state move is $980. This average cost includes both do it yourself (DIY) movers and Americans who hired professional movers. For many, this means that hiring professional movers is not feasible because of the high cost (Wood, 2020). Moving comes with many obstacles in addition to the cost—from packing boxes and safely loading them onto the truck to the potential loss or misplacement of important items are when a move is finished.

All these problems culminate in two major issues for Americans: moving is expensive and tedious. To help solve these issues, an application for planning a DIY move is a solution that saves the cost of hiring professional movers and saves time by making the planning more efficient. Load.In solves these problems by providing a smartphone application for DIY movers to plan, catalogue, and gain expert knowledge on how to move more efficiently. Load.In allows the user to take pictures of their boxes and through photogrammetry produce 3D models of those items, access expert tips and tricks on how to pack and load a variety of items, and view a step-by-step packing solution to guide the mover through every step of the process—all while keeping track of where each item is during the move.

1.1 Purpose

Load.In is a mobile application that can be used by any person or group of people to plan and execute a move. Load.In uses photogrammetry along with user input to generate a list of boxes that a user plans to move. This list is then fed to the loading algorithm which takes the dimensions, weight, and fragility to load the boxes most efficiently onto the truck. Load.In is an application that gives DIY movers the ability to plan and move like a professional. In addition to

the measuring and loading of items, Load.In contains a tips and tricks section to help users with packing and loading their truck. Load.In also features an inventory system that keeps track of a user's boxes and items contained within for the entire move. The boxes are identified in a user's inventory by an identifying integer that begins with one for each user. Using these identifiers Load.In keeps track of where a user's box is at any given time. This could be already loaded on the truck, arrived at the destination, or not loaded yet. These features allow users to optimize the cost and time it takes for their move by planning around different truck sizes. This allows users to plan for more trips in a smaller truck or less in a larger truck.

The goal of Load.In is to give DIY movers the power to move themselves rather safely and efficiently than hiring a professional moving company. By keeping track of all a user's items and planning the most efficient way to load their boxes, Load.In will help users plan and pack a load effectively. Another goal of Load.In is to effectively design a scalable software solution to this problem that potentially could apply to more than a DIY mover. The ability to change container size and box sizes give the application the potential to be used for far more moving problems, such as packing planes, or ships. Load.In is an all-in-one solution to a DIY move. Keeping track of items, generating a plan to load and unload the truck, and providing tips on how to pack makes the move much more efficient. Load.In provides DIY movers an efficient move without the cost of hiring professional movers.

1.2 Scope

Load.In provides capabilities for users to plan their move. The biggest feature is the 3D load plan generated on the app. This works through photogrammetry— converting images of a user's items into 3D models. With each picture the app determines whether an accurate model can be generated based on the item's dimensions, and if not, it has the user take another picture

from a different angle. Once unique items are photographed and modeled, the algorithm takes these 3D models and prepares the load plan. This algorithm considers the fragility of items as well as weight and size to determine the safest and most effective way to load the items. This generally means that items like furniture are packed first and other items stacked on top of them. The load plan suggests a truck size and directs the user to where such a truck can be rented. If the user specifies a truck size or chooses a different truck, the plan is regenerated for that new vehicle. The load plan is meant to be a step-by-step manual on how to load and unload the truck. It specifies a location and order of how to pack items to keep them safe. This also means the load plan keeps track of a user's inventory displaying where items are within the truck.

With the user taking pictures of all their items, Load.In maintains a log of all the items. This means that the user will have a list of which box has which items. The utensils the user packed are labeled as designated by the user—meaning they can take that box straight to the kitchen in their new home. This inventory means the important items that users may need right away, such as a computer, will be easy to find and unpack.

In addition to the inventory, Load.In has a cost estimation feature generated from the information the user provides. By taking the load plan, the truck size, and start and end locations of the move, the app calculates an estimated total cost for the move. Load.In works with rental truck vendors to provide the best estimates and deals to its users. This is done through scraping the web and pulling in data from the rental companies to gather truck sizes, costs, and availability. This allows Load.In to show users the best options that are closest to them, and update in real-time as more trucks become available. With all of these factors, Load.In shows a tangible cost benefit to the user over hiring professional movers.

In order to help a user begin boxing their items, Load.In provides expert tips and has a live chat. These two services help with packing heavy, oddly shaped, and fragile items to keep them safe. The chatbot and live help are used to either guide a user to articles about the topics they want help with or provide live interactive service to the user. This is for questions as to how the app itself works or something as simple as removing the legs of a couch. The tips and chatbot are there to improve the user experience with quick helpful tricks.

Load.In uses user data to generate analytics on DIY moves throughout the country. The important pieces of information that Load.In gathers include location data, meaning a move's start and end locations, and the total amount of trips and time it took a user to complete that move. Data on the move itself is collected including the cost of gas, the cost of renting the truck, the cost of any other supplies such as boxes, as well as the time spent loading and unloading the truck. The specifics on a user's inventory are kept until the move is completed, and the less sensitive data will continue to be stored. Data such as how many fragile items were packed, the weight of the entire move, and the dimensions of the different items or boxes are stored. Data will be used to generate predictive analytics about the overarching meaning of these statistics. Generating predictions on where people are moving from and to, how far they are moving, and how much they are moving are valuable assets that can be either sold or used to make a more efficient service to Load.In users.

The final functions of Load.In are to gather feedback from the user. This is rating their experience with Load.In and how it helped their move. Giving feedback on the move helps give a direction to improve Load.In for future users. Another way in which Load.In works to improve the application is using a heat map to track what users are accessing the most. It tracks taps on

the app as well as when the app is closed so Load.In knows exactly what features are being used the most or the least and what needs to be improved upon.

Load.In is being developed for DIY movers in America. The goal is for the app to be useful for a move just across the street or a move 3,000 miles across the country. The case study for Load.In is a single family of 3 with a dog moving less than 20 miles across town. They need to move the contents of an approximately 2,200 square feet household which is the average house size US Census Bureau (2019). This move also needs a rental vehicle as it has too many items for just the family vehicles. A few of the mandatory items involved in the move are furniture, books, and silverware to get a more accurate picture of a typical move. Load.In needs to accurately tell the costs of this move and allow friends or family to collaborate through the app on the move. Really odd shaped items are left out of this case study. The final goal for the Load.In case study is to have the app work well for a common local move but be able to see that it is scalable. This means determining if the application is just as effective with a variety of different move sizes.

[This space is intentionally left blank]

1.3 Definitions, Acronyms, and Abbreviations

3D – A three-dimensional form or appearance.

Administrator – Someone who will access elevated features of the Load.In system in order to maintain and detect issues.

Amazon Lambda – A serverless compute service that allows code to be run without the need for provisioning or managing servers.

Amazon RDS – A distributed relational database service provided by Amazon Web Services.

AWS – Amazon Web Services: a cloud platform on which Load.In's databases are hosted. **Android** – A mobile operating system based on a modified version of the Linux kernel and other open-source software.

Android Client App – The client-side application for Load.In which runs on the Android platform.

API – Application programming interface: an interface for programs to share information and functionality with one another through a series of calls or connections.

AWS Elastic Beanstalk – An orchestration service offered by Amazon Web Services for deploying applications which orchestrates various AWS services including EC2, S3, Simple Notification Service, CloudWatch, autoscaling, and Elastic Load Balancers.

AWS Elastic File Storage – An AWS service that provides file storage with the ability to auto scale up with increased demand.

Apache CFX – A popular library for hosting web APIs.

Apache Tomcat – An open-source implementation of the Java Servlet, JavaServer Pages, Java Expression Language, and WebSocket technologies. Tomcat provides a "pure Java" HTTP web server environment in which Java code can run.

Chatbot – A feature within Load.In that provides information to users and guides them towards helpful articles and other resources interactively.

Cloud – A term used to describe several computing models such that a company or individual can purchase resources for hosting a variety of things in a centralized location accessible from anywhere in the world.

Computer Vision – a subclassification of artificial intelligence that involves computing information about the world from various sensory data such as images. Techniques of this classification are used throughout Load.In to observe real world objects.

Container Loading Algorithm – A type of algorithm that attempts to optimally fill a three-dimensional space with physical objects. Load.In uses this kind of algorithm to generate Load Plans.

CPU – Central processing unit: the primary component of a computer that processes instructions.

CSS – Cascading style sheet: a style sheet language that is used for formatting the layout of Web pages.

Do-It-Yourself (DIY) Mover – Non-professional movers who rent a truck for their move, and handle all packing, unpacking, and manual labor themselves. This is the primary end user of Load.In.

Expert Tips – Feature of Load.In that allows for a mover to search for helpful articles pertaining to a variety of useful information on how to accomplish various tasks during a move.

GHZ – Gigahertz: a commonly used unit when measuring computer processing speeds.

Guest – Someone who is accessing the Load.In system anonymously and has not registered for an account or someone who has registered but has not authenticated to the system at the time of access.

GUI – Graphical user interface: the aspect of a software program that the end user interacts with.

HTML5 – Hyper Text Markup Language version 5: a markup language used for structuring and presenting content on the Web.

Heatmap – A data visualization technique that shows magnitude of a phenomenon as color in two dimensions.

Java – A set of computer software and specifications that provides a system for developing application software and deploying it in a cross-platform computing environment.

JavaScript – A scripting language that runs in the browser and performs one or more functions to animate an otherwise static HTML document.

Linux – An open-source and community-developed operating system for personal computers and workstations.

Load Plan – A set of instructions on how to optimally load a container – generated automatically by Load.In from the boxes and furniture input into the system by the user.

Logistics Planning – A feature of Load.In that assists the mover with determining what rental trucks cost, how many trips the truck might need to take and whether the truck is available to rent based off proximity to the mover.

Mbps – Mega-bits per second: a unit of measurement for network speeds.

Megapixel – One million pixels: typically used to measure the size and quality of images.

Move Analytics – A feature of Load.In in which information gathered from previous moves are used to determine estimations for future moves as well as predict market trends for Rental Companies.

Move Inventory – A feature of Load.In that catalogs all boxes and items the mover intends to move.

MySQL – An open-source relational database management system.

MacOS – An operating system used on Apple's MacIntosh line of personal computers and workstations.

OS – Operation system: a collection of programs designed to provide a platform on a device to run other applications and typically provides a layer of abstraction from the hardware it interacts with.

Pixel – A small square of color that is part of a larger display screen or image.

Photogrammetry – A computational method of deriving three-dimensional information from images. This method is used in Load.In to construct 3D models of boxes, furniture, and other items from pictures taken from the end user's cell phone camera.

PNG – Portable Network Graphics: a common image file format that Load.In uses.

Professional Mover – Professionals who handle the physical labor of loading and unloading a moving truck as well as driving the truck to the destination.

Rental Administrator – A representative of a rental company who will access the Load.In system on behalf of the rental company.

Rental Company – Any company that rents moving vehicles for a Do-It-Yourself Mover to assist them with their move.

Rental Estimate – A feature provided by Load.In that pulls data from the internet to determine the cost of renting a moving truck.

Smartphone – A device, typically handheld, that can act as both a cellular phone and a computer by running one or more applications typically through a touch screen interface.

SPRING MVS – An application framework and inversion of control container for the Java platform.

Test Harness – A set of special features used during the development of Load.In to enable testing and demonstration of the application.

Vendor Synchronization – A feature of Load.in that brings in truck sizes and availability of rental information from third party moving company websites.

Windows – An operating system developed by Microsoft for use on personal computers and workstations.

[This space is intentionally left blank]

1.4 References

Andrew, P. (2020, January 26). Is your house the "typical American home"? *Hsh.* https://www.hsh.com/homeowner/average-american-home.html

CADCode Systems. (n.d.). *Optimizing & machining.* CADCode. Retrieved September 20, 2020, from https://www.cadcode.com/category/categories/optimizing-machining

Collins, T. (2018, April 20). A look into photogrammetry and video games. *Medium*. https://medium.com/@homicidalnacho/a-look-into-photogrammetry-and-video-games-71d602f51c31

Dube, E. (2006). *Optimizing three-dimensional bin packing through simulation.* Semantics Scholar. https://www.semanticscholar.org/paper/OPTIMIZING-THREE-DIMENSIONAL-BIN-PACKING-THROUGH-Dube/bb9986af2f26f7726fcef1bc684eac8239c9b853#references

Economy Moving & Storage, LLC. (2015, January 4). *How to properly pack and load a moving truck- movers Cincinnati* [Video]. YouTube. https://www.youtube.com/watch?v=rjmofUZOdwo&feature=youtu.be

Knoblauch, M. (2019, May 8). One in ten Americans would prefer a week in jail over moving. *New York Post*. https://nypost.com/2019/05/08/one-in-ten-americans-would-prefer-a-week-in-jail-over-moving/

Manwaring, K. (2020, November 24). *The average cost of moving truck rentals.* Move.org. https://www.move.org/average-cost-truck-rental/

Meyers, S. (2018, October 10). *How much does it cost to move a four (4) bedroom house?* Movers.com. https://www.movers.com/moving-guides/moving-four-bedroom-house-costs.html

Nat and Friends. (2017, April 18). *Google Earth's incredible 3D imagery, explained* [Video]. YouTube. https://www.youtube.com/watch?v=suo_aUTUpps&feature=youtu.be

Team Yellow. (2021, February 28). Lab 1 – Load.In Product Description. Retrieved March 28, 2021 from https://www.cs.odu.edu/~411yello/deliverables/labs/cmiller_lab1_V2.pdf

The American Institute of Stress. (n.d.). The Holmes-Rahe Stress Inventory PDF. Retrieved September 20, 2020, from https://www.stress.org/wp-content/uploads/2019/04/stress-inventory-1.pdf

US Census Bureau. (2019, October 10). *Historical households tables.* The United States Census Bureau. Retrieved September 2020, from https://www.census.gov/data/tables/time-series/demo/families/households.html

White, M. (2018, May 10). *How to pack & load a moving truck.* Moving.com. https://www.moving.com/tips/loading-truck-rental/

Wood, P. (2020, November 4). 29+ Moving industry statistics. *MoveBuddha.* Retrieved November 20, 2020, from https://www.movebuddha.com/blog/moving-industry-statistics/

Yale, A. J. (2019, March 28). *Report: Where Americans are moving - and how far they're going.* The Mortgage Reports. https://themortgagereports.com/49116/report-where-americans-are-moving-and-how-far-theyre-going

1.5 Overview

This product specification details the software and hardware components, features, and external interfaces of the Load.In prototype. The information in the remaining sections offers a detailed description of the software and hardware being used to implement the Load.In prototype, the features the prototype contains, limitations the prototype has, and the interfaces that Load.In uses.

2. Load.In General Description

Load.In is a software based solution to loading items for a DIY move. This involves an inventory of boxes, expert tips on how to pack items, photogrammetry for measuring boxes, loading algorithm to plan a load based on the inventory, and a 3D rendering of the plan step by step. The application links a user's inventory to their account through a user ID which is how the application keeps track of who is logged in at any given moment. Each function in the app checks the user ID and provides only the information on their move. The user has the ability to add boxes to their move and edit values for that box. These values include a short description of what the box contains, the dimensions of the box, the weight on a scale from one to five (five being the heaviest), fragility on a scale from one to five (five being the most fragile), and a list of items contained within the box. These boxes will populate a user's inventory which is what is used to generate their load plan.

2.1 Prototype Architecture (Hardware/Software)

The Load.In prototype is being developed on a virtual machine with Ubuntu Linux operating system. The application makes use of Docker containers that include Apache Tomcat, CFX for the web API, and MySQL. The user interface will be an Android application, and the prototype will also have a Test Harness interface. Tomcat is going to be used as the platform for the web

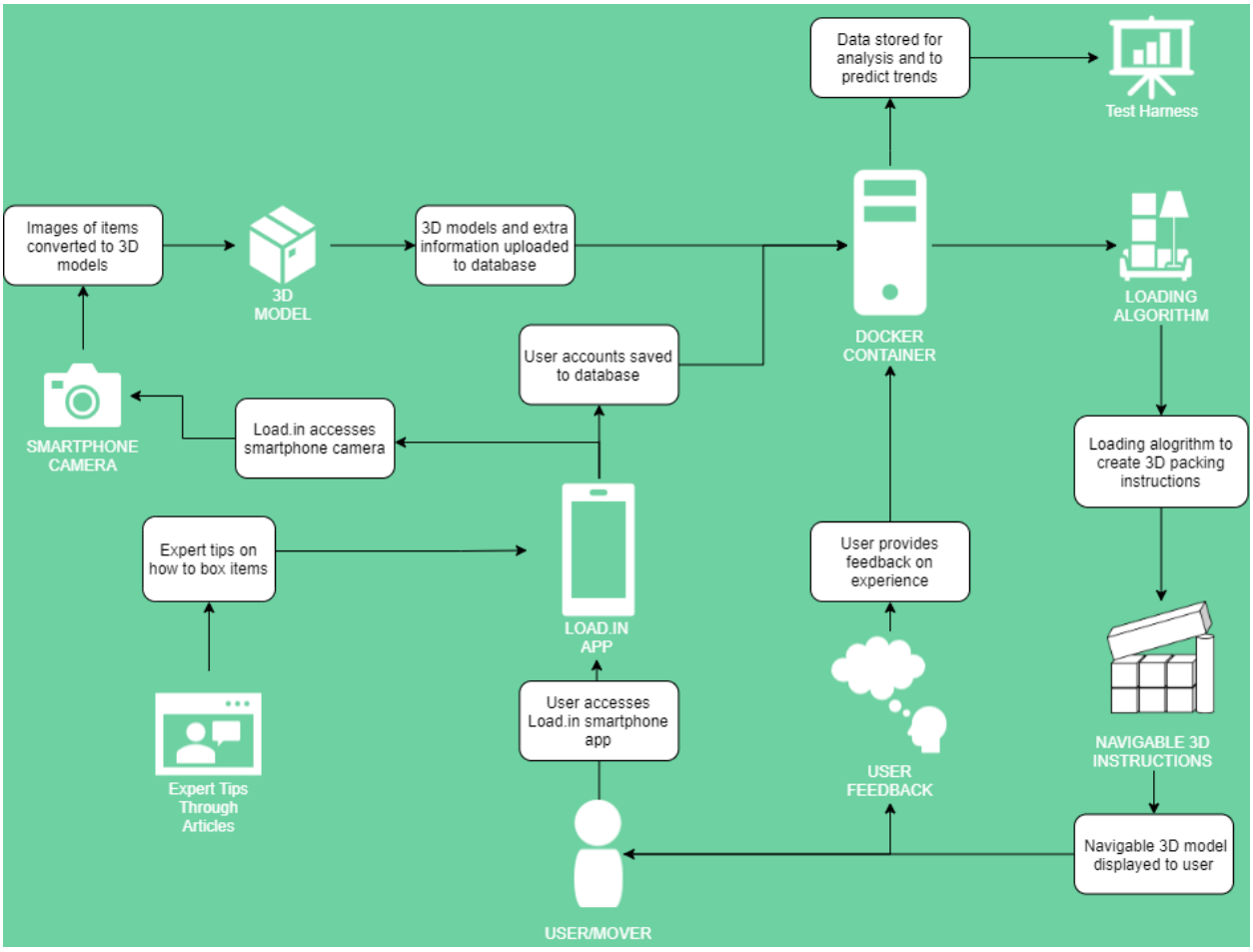
API, and MySQL will be the platform for maintaining data. The app itself will be an Android application which is how fake clients will interact with the other systems. The test harness will be a built-in way to create many different testing features as to ensure the functionality of the application. The Test Harness will include moves of different sizes, including one based around the case study. It will be useful in demonstrating the scale and scope of Load.In.

Due to the differences between the Load.In prototype and the real world product, the Major Functional Components Diagram shown in Figure 1 has been cut down to reflect those changes. Many of the functions have either been fully removed or will be implemented in a limited fasion in order to show a proof of concept and nothing more. This includes the removal of some functions such as the heatmap and the reduction of other functions such as the expert tips. The components that are being included in the prototype are shown in Figure 2.

[This space is intentionally left blank]

Figure 1

Load.In Prototype Major Functional Components Diagram



One of the major goals of the prototype is to demonstrate its effectiveness against the case study mentioned in section 3. Being able to demonstrate a move plan for this family will be a proof of concept showing that Load.In will be effective in its mission of helping the DIY mover.

2.2 Prototype Features and Capabilities

The Load.In prototype has features in order to demonstrate a proof of concept. The prototype will demonstrate the ability to measure an item through photogrammetry and generate that item into a 3D model. The data will be kept in an inventory that the user can access to add, delete, or edit items. The load plan is generated from these models and keeps track of where the

items are located within the truck. The load plan which will guide the user step-by-step through loading the boxes properly onto a truck. The prototype will also include an estimation for how many trips a user will need based upon the inventory and truck space available. The last two features the prototype will include are expert articles and a feedback feature. The changes from the real-world product to the prototype are demonstrated Table 1 mapping how the functionality changed.

Table 1

Load.In Prototype Features Table

Feature	**Description**	**Implementation**
	Authentication	
Login User Interface	Create a landing page for the login screen.	**Full Functionality**
User Registration	Create an account inside of the load.in application.	**Full Functionality**
Reset Login	End user to be able to reset his/her own password inside of load.in application.	**Partial**
User Login	Authenticate with the server that an account exist.	**Full Functionality**
DIY User Interface	Create a user role for the end user with normal privileges.	**Full Functionality**
	Move Inventory	
Furniture/Item Measurement	Ability to measure items using the phone camera.	**Partial**
Move Inventory Interface	The interface shall consist of multiple different screens to manage the inventory.	**Full Functionality**
3D Model Generation	Generate a 3D Model based on item measurements	**Partial**
Box Locator	Users shall be provided with a search bar on the Move Inventory Interface to allow	**Full Functionality**

	them to search for a specific box by the box number or description, they will be provided with the location of the box.	
	Expert Help	
Expert Tips User Interface	The user interface for expert tips shall alert the end users when an expert tip is found.	**Full Functionality**
Expert Articles	The expert articles shall provide useful information to the end user based on keywords stored for each article.	**Full Functionality**
	Load Plan	
Generation Algorithm	The Generation Algorithm for the Load Plan solves placements of Move Inventory items in a given truck.	**Partial**
Load Plan Display Interface	The Load Plan Display Interface takes a given Load Plan and displays it to the user.	**Full Functionality**
	Logistics Planning	
Rental Truck Costs	Calculate the cost of renting a truck.	**Partial**
Move Estimates	Calculate the number of trips one will take based on the truck size.	**Partial**
Move Inventory Storage	The system shall provide a mechanism by which the user inventory will persist and be accessible anytime by the user.	**Full Functionality**
	Analytics	
Feedback data	Ability for the end user to provide feedback for a move.	**Partial**
	Test Harness	
Sample Move Inventory	The test harness shall include a preset move inventory that will be representative of an	**Full Functionality**

	average move in the United States.	
New Truck Size	The New Truck Size feature in the Test Harness will allow for testing of the Load Plan with different sized trucks.	**Partial**
	Algorithms	
Container Loading Algorithm	This algorithm works on the back end to generate a load plan that minimizes space of the user's moving truck given a set of boxes.	**Full Functionality**
3D Model Generation	This feature will be utilized in order to efficiently capture real-world measurements of boxes, which will later be used as input into the Load Plan algorithm.	**Full Functionality**
Expert tips Algorithm	The algorithm shall take in keywords based on the end users input for box content description during the move inventory phase.	**Full Functionality**
	Database	
Relational Database	The relational database shall store all persistent application data.	**Full Functionality**
User Credentials	The system shall provide a mechanism by which the user credentials can be persisted and retrieved at any time for the purposes of authentication.	**Full Functionality**
	Web API	
Authentication	The web API shall provide both a mechanism for authenticating users to the android client but also to authentication the user when utilizing services for the web API.	**Full Functionality**

Password Rest	The web API shall receive a request to reset the password from the android application.	**Full Functionality**
Expert Tips Indexer	The web API shall provide a mechanism for indexing the mySQL database that contains the Expert Tips.	**Full Functionality**
Services	The web API shall provide various services for handling RESTful requests made from the Android application.	**Full Functionality**

These are the core features to what makes Load.In a unique solution to the problems of DIY moving. Some of these features, such as the expert tips, will be a proof of concept. Load.In will have features that are meant to mitigate some of the risks involved in developing such an application. The feedback feature is meant to mitigate the risk of customers being unsatisfied with the aspects of the app from how it measures items to how it plans a user's move. Another risk is the user not following the guidelines of the application. Loading boxes out of order or leaving items behind intentionally can alter how a user interacts with Load.In. To mitigate issues that could arise from such, changes Load.In will be able to alter a plan based on changes a user makes. The security risks such as the pictures taken by the user and the security of the user's data will be mitigated by the prototype storing photos locally on the phone. The data will also only contain information for the purpose of the move. A user's data will also be able to be deleted at any time through the Test Harness.

The goals of the Load.In prototype are to prove that the application can work in a real-world setting by providing expert tips and a plan to move from one location to another. This will be done through capturing a move inventory, generating a load plan based on that inventory, being able to change truck sizes, generating and searching for expert tips, and the ability to

search for items within a load plan. In order to test many of the features, the prototype will be able to generate a random inventory so there can be a variety of test cases.

2.3 Prototype Development Challenges

The main development challenges involved with Load.In specifically involve the many variables that will go into a move. This includes the Test Harness changing values in the database, changing the truck sizes, changing the move inventory, prioritizing results for the expert tips search, determining and setting multiple trips, and the handling of edge cases. The handling of edge cases is probably the most difficult when it comes to this application. This involves ensuring a user cannot overpack a truck or have boxes that are too large for the truck itself. There are potentially problems with the boundaries of each variable and without careful implementation, the prototype could fail to run correctly. Other development challenges are balancing the use of already existing open-source software versus development from the ground up. Balancing these is the key to keeping Load.In innovative, and also keeping development on schedule.

2.4 External Interfaces

Load.In uses a few different interfaces throughout the prototype. There are networking, software and hardware interfaces that the prototype works with to provide the user the solution. The networking interfaces include TCP/IP to establish a network connection between the smartphone application and the web API. The application will also use standard DNS and DHCP protocols in order to establish and maintain connections. The application will use SSL to keep connections secure between the app and the web API. In terms of hardware, Load.In will use a smartphone with a camera to take advantage of photogrammetry. The phone will have to install an apk to use the Load.In application. The prototype will have access to a few generated users

within the database that have different inventories to have quick and easy access to various moves to show functionality of the application efficiently.