

Lab 1 - Load.In Description

Lance Perdue

Old Dominion University

CS 410

Professor Janet Brunelle

31 January 2021

Version 2 - Draft

### Table of Contents

1. Introduction.....	3
2. Product Description .....	4
3. Case Study .....	7
4. Product Prototype Description.....	8
5. Glossary .....	13
6. References.....	16

### List of Figures

Figure 1 – Load.In Major Functional Components Diagram.....	7
Figure 2 – Load.In Prototype Hardware and Software.....	9
Figure 3 – Load.In Prototype Major Functional Components Diagram.....	10

## 1. Introduction

The problem when it comes to DIY movers is that one lacks the expertise to handle the logistics a move. In a study conducted by OnePoll, the average American takes four and a half months to unpack after moving (Knoblauch, 2019). The cost of moving can increase exponentially when choosing between professional moving companies or a do it yourself move (Wood, 2020). Keeping track of everything when packing can pose a problem when moving. Misplacement of items is stressful when the DIY mover must search through every box to find a birth certificate. Subsequently, DIY mover are not utilizing using the space of a moving truck effectively. This can lead to the DIY mover making more trips than necessary. More trips the DIY mover must compete means more money out of the DIY mover's pocket. According to a study by the New York Post, more than one in ten people would rather spend a week in prison and than move (Knoblauch, 2019).

One has two choices when it comes to moving: Hire a professional moving specialist to take over the move or conduct a do it yourself move. The problem with professional moving companies is that it can quickly become expensive. The cost of hiring a professional moving company can vary depending on the distance of the move. A move between the states can be expected to cost more than \$4,300 (Wood, 2020). For most people, this price is unaffordable. For a move within the state, a payment of over \$2,300 can be expected (Wood, 2020). If one decides to act without the assistance of professional movers, one will find that the cost is less expensive. If someone decides to move locally, for example, can expect to pay between forty and seventy US dollars per day for a rental truck. These costs are manageable for moving around the

city but can be inefficient due to lack experience. Since time is money, that time could be made up by using professional movers. When it comes to running a long-distance move, one can pay more than \$1,800 to \$2,000 to rent a 10-day truck for one or two bedrooms. 78.3 percent of the moves in the United States are DIY moves, with over 31 million people moving each year (Andrew, 2020). With 82.7 percent of people conducting a move within their home state, the number of rental trucks used is a market Load.In can make an impact in (Andrew, 2020).

Load.In is an implementation of the move process that gives ones move a game plan. With Load.In, one will be able to catalog boxes, measuring furniture, estimating truck sizes, know the number of trips a move takes, packing like a pro and knowing where all items are. With the assistance of artificial intelligence and computer vision, one can become an expert in moving. The use of artificial intelligence will enable the end user to place objects professionally in the moving truck. Using computer vision, the sizes of the boxes are measured and displayed to visualize the placement of the boxes. Helpful tips and tricks are also provided to the end user to help with moving. If the end user needs help moving an object, the end user will also receive expert-level instructions. Load.In is the solution for the DIY person to speed up the relocation process and turn one into a professional.

## Product Description

Load.In provides a load plan by providing step-by-step instructions to solve the logistics of moving. The load plan will consist of the collection of data from using computer vision to measure boxes and furniture. To take advantage of the computer vision, the end user will take photos of the objects that will be loaded into the truck. The end user must take enough photos of the object to create a 3D model, and the 3D model is then stored in the database. Each scanned article is assigned a QR code, which points to the position of the article. Any image left over

after rendering the 3D model is then deleted from the system. The load plan then shows the end user where each item should be placed in a moving truck. The load plan will utilize the truck space available against the 3D rendered models left to load. Each object will be processed to determine the placement inside of the moving truck.

End users can estimate the cost of a move by indicating the amount of trips, the estimated time of a move, and the cost of the truck. The number of trips a user will take will depend on the items the user has scanned into the system minus the truck space available. By collecting more moving data from end users using the application, Load.In can estimate the moving time for a person and estimate how much a truck rental will cost.

Another important feature that Load.In will offer end users are expert tips. Packing can be a stressful event if someone has no experience in moving. Load.In gives tips and suggestions on how to pack certain items. Consumers can enter the item they are having problems with and receive instructions on how to package it. The instructions will be written by our moving expert to provide professional moving advice.

If expert tips do not help the end user, they can connect to a chatbot for additional assistance. This live chat bot operates with artificial intelligence to answer any questions that are asked. The chatbot will respond to the user with tips that are stored in the database when a key word is entered. If the chat bot is unsuccessful in answering questions, the chat bot falls back to a live representative.

Web scraping is used to collect data from seller websites such as truck rental companies. The data collected from the websites is the availability of trucks and rental costs. This data is then stored in the database in order to inform the user about current prices and truck rentals at the rental company.

Currently the amount of data when it comes to moving is limited. One of the features of Load.In is to provide the analytics of a move. The analytics collected by the application are location, cost of move, user inventory, and feedback. The recorded location data includes the distance traveled, number of trips, and the start and end point. The relocation data includes the recording of the gas, rent, and supply costs of a move. The collected inventory data of the user are weight, fragility, dimensions, and box dimensions. At the end of a move, Load.In collects the user's feedback to improve the application. The collection of all this data is summarized in a report made available to the vendors who pay for access. This type of historical move data has never been collected before and will be a major selling point for Load.In.

Load.in will utilize a heat map that will help identify problem areas in the user interface. Load.In will track which areas of the user interface are not being utilized the most. For this purpose, Load.In implement a heat map which records event-based data and stores it within the database. This heat map will also have the ability to tell when a user has stopped using the application. This will help improve the next iteration of the application to fix the areas flagged in the heatmap.

The database in which these functions are stored will be a relational database. Amazon RDS and EFS are used to remotely store the data for this application. The most important database tables that are needed include user, expert tips, rental information, and analysis. To manage the entire database, the preferred language will be MySQL. The user's device also stores the photos at the local level to save space and protect the user's privacy. The application runs on Android smartphones running version 4.4 KitKat and higher. The CPU needs eight cores, which operate at 1.8 GHz. The Internet speed on these devices must be at least 15 Mbps to transfer data to the server. The required amount RAM to load.in properly is 4 GB. The smart phone device is

used to capture the measurement of the boxes and furniture. This data will then be stored in the database to create a move plan.

The website client page runs under Linux, Windows and macOS. Since it is only a web browser, most applications can connect to the web client. Important web browsers like Firefox, Chrome, Edge, and Safari that support ES6 or higher will be supported. The internet speed must be 30 Mbit / s. The amount of RAM required to run load.in properly is 4 GB.

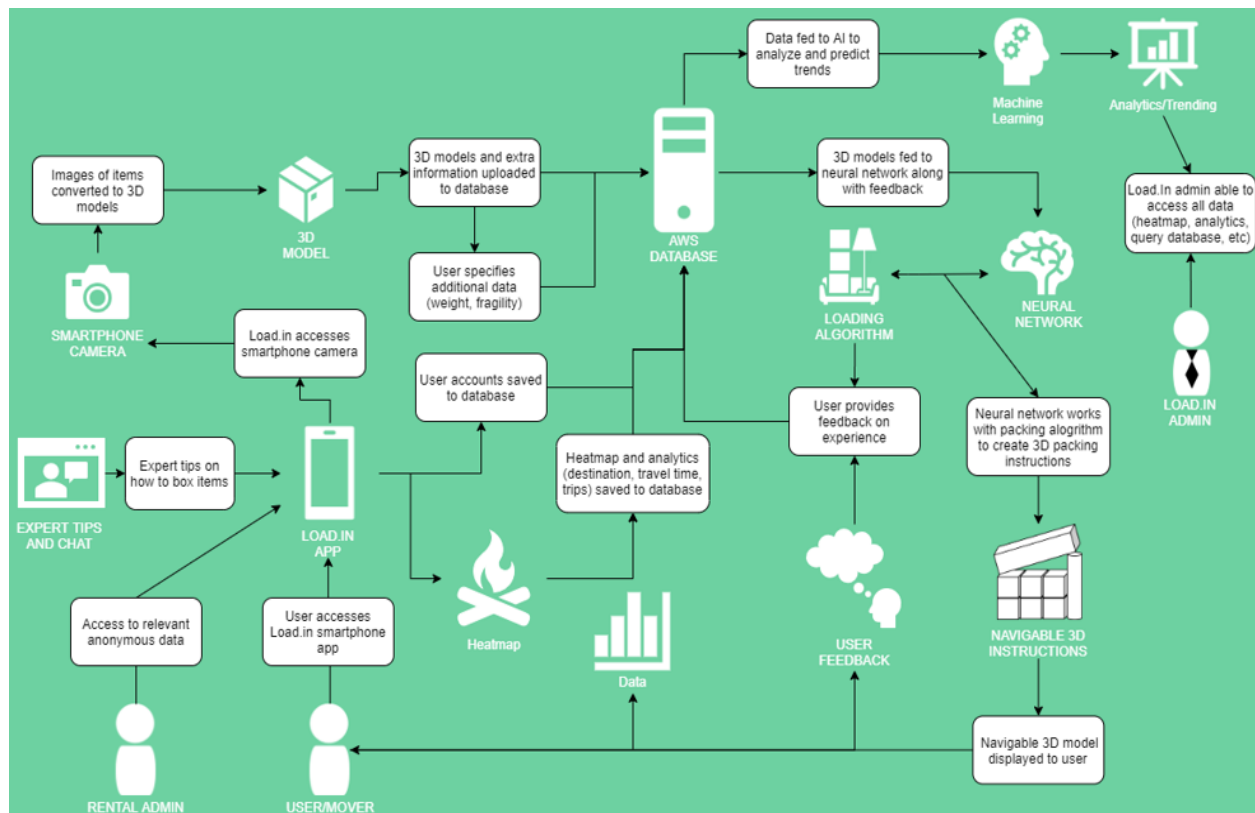


Figure 1 – Load.In Major Functional Components Diagram

## Case Study

Load.In is an application designed to help people move across the county. The case study for Load.In is an average family of three with a dog. A family of three is the average household according to the United States Census Bureau from December 2020 [11]. Getting this

average family in the prototype phase will help target a wide array of issue that could occur during development. This family will have a decent amount of furniture to move, requiring a moving truck to be utilized. The 3D model generation is utilized on the furniture to generate a load plan. When the load plan is generated, the family will then tell the cost of the move and how to load like a professional. In future iterations of the application, Load.In could potentially expand to the industrial industry by offering efficient loading.

## Product Prototype Description

The prototype for Load.In will demonstrate a proof of concept for the application. Some features discussed are reduced or removed completely. The measurement of boxes, 3D model generation, load plan, and feedback will be reduced to partial implementation. The box locator, estimated trips, packing tips, tips search, and expert articles are going to have full functionality. The features that will be removed is weight, fragility, unloading instructions, chatbot, live expert, web scraper, vendor web API, truck sizes, truck availability, location data, move data, heatmap, and rental interest statistics.

To bring this project to live, some key components are needed for a smooth development cycle. Load.In development is running on an Ubuntu 16.04 virtual machine. To bring all of the components together, Docker is installed on the virtual machine. Docker runs the Apache Tomcat, CFX for web API, and MySQL. Android Studio is the development environment Load.In will operate on. IntelliJ is used for all other developments outside of the Android



application.











Category	Android App	Web API
Version Control		
Issue Tracker		
CI/CD		
Language		
Platform	 Android Native App	 Apache CFX on Amazon Elastic Beanstalk with Tomcat
Database	 (or Aurora) on AWS RDS	
IDEs	 Android Studio	 IntelliJ
Developer OS	 Linux	

Figure 2 – Load.In Prototype Hardware and Software

Docker is used utilized as a platform to bring most of the key components together. Tomcat is used to host the web API in order for the application to communicate with the database. MySQL is used to develop the database when the end user utilizes the application. The end users are interaction with the Android application. This is either with a physical phone or built in SDK emulator that came with Android Studio. The test harness changes values inside of the database when demonstration the applications proof of concept.

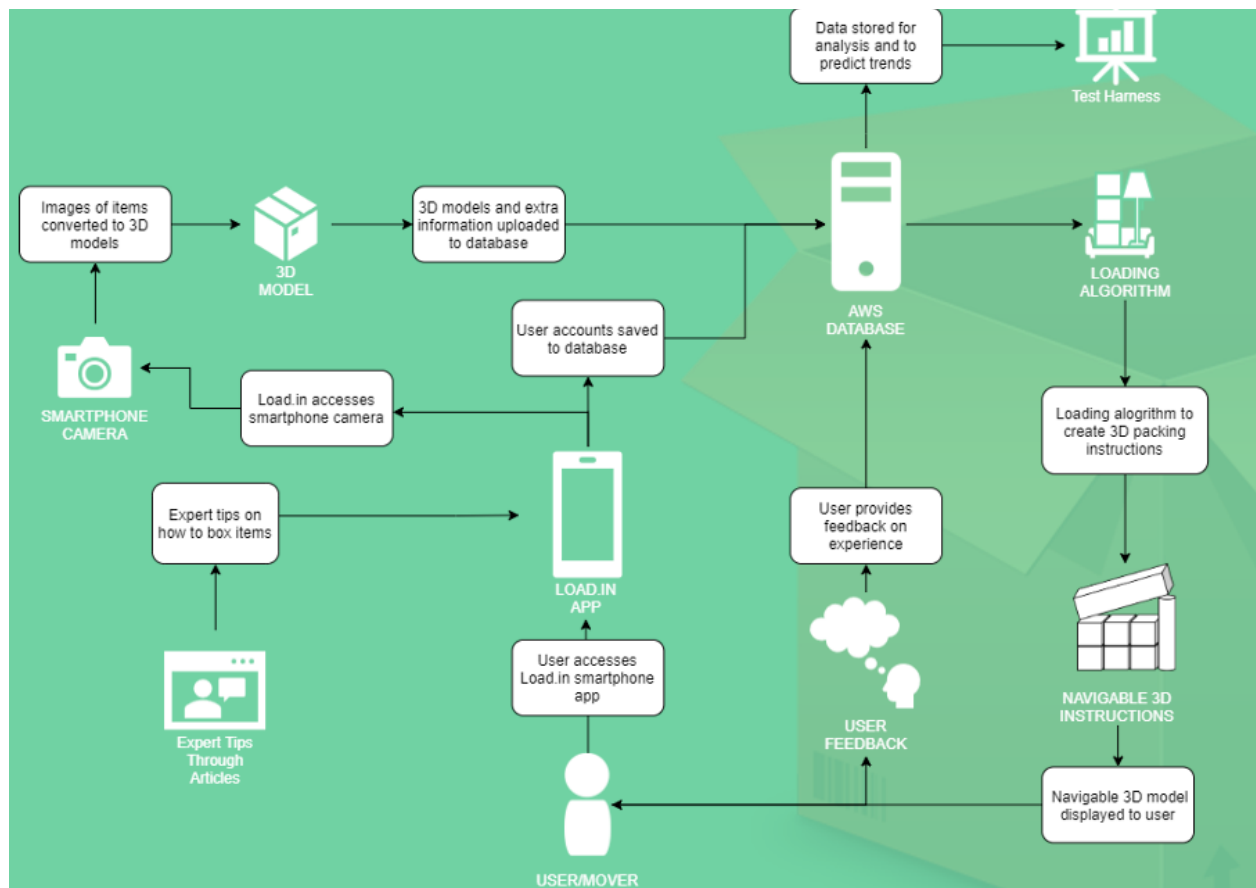


Figure 3 – Load.In Prototype Major Functional Components Diagram

The prototype features that are kept demonstrator a proof of concept for the application. Measuring an item using the phone camera is demonstrating how items can be measured and stored into the database. If the phone camera can't be utilized, the measurements are to be entered manually. The measurements entered into the application are then used to make a 3D model. Each item scanned in will have a QR code to identify each box location. The load plan feature will configure how each of the boxes will be oriented inside of the truck based on the available free space. This is innovative because an item placement algorithm has the potential of expand beyond loading trucks. Next, the estimated number of trips you make is also calculated based on the free space. If the truck space is filled and there are still boxes left, number of trucks increases by one each time. When an end user runs into an issue and doesn't know how to pack

an item, packing tips are made available for the user to search. The end user can type in an item name and find expert level articles based on the item typed in. After a move is completed, the end user can provide feedback on how the move went. This will help Load.In improve the application over time with the provided feedback.

With each project risk mitigation is needed. Some end users will not be satisfied with the recommendations the application makes. To mitigate someone not liking the application, customers have the ability to provide feedback after a move to disclose any issues with the application. Another customer risk is if an end user doesn't follow the guidelines of the application. The mitigation is to implement a feature during the load plan that allows the user to repeat certain steps in the application as they progress in the move. A security risk Load.In can run into is the private photos taken on the phone to generate the 3D model of the boxes. The end user doesn't want these photos to fall into the wrong hands. To mitigate this security risk, Load.In will not upload personal photos to the server. The final security risk is if the end user wants to ensure the data collected isn't used for nefarious purposes. The mitigation is to allow the end user to delete the move data at any time.

The goal of Load.In is to show a proof of concept that a fully functional application is worth devoting time to. Capturing the end users' inventory in order to generate the load plan is critical. Load.In goal is to use the camera to capture the measurement. If this is not possible, then manually entering the dimensions is the next step. Goal for the load plan is to demonstrate multiple plans based on the items entered into by the end user. The load plan must adapt if the size of the truck is changed. Expert tips need to be able to find tips based on the items searched. During a move, the end user must know what is inside of each box scanned in.

Along the way multiple challenges could occur. The test harness changing values in the database is a major function to prove the load plan actually works. The challenge for this is implementing a random number generator for measurements to override the current values the end user placed in the database. Also related to the load plan is changing the truck side. This must be accounted for in an event the end user arrives at a rental station and the truck in the application isn't available. Prioritizing search results when an end user search for tips could pose a challenge if the wrong article is shown first. That could easily render this feature useless if the end user types in piano and all the results show fine china. The biggest challenge Load.In could face are the edge cases that were not thought of during development.

## Glossary

**3D** – Three Dimensional

**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 lets you run code without provisioning or managing servers

**Amazon RDS** - Amazon relational database service

**Amazon Web Services (AWS)** – 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

**Application Programming Interface (API)** – An interface for programs to share information and functionality with one another through a series of call 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.

**CPU** – Central processing unit.

**CSS** – Cascading style sheet.

**Do-It-Yourself (DIY) Mover** – Non-professional movers that rent a truck for their move, but and handle all packing, unpacking, 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

**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

**Java** - a set of computer software and specifications developed by James Gosling at Sun Microsystems, which was later acquired by the Oracle Corporation, 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 function to animate an otherwise static HTML document.

**Linux** - An open-source and community-developed operating system for personal computers and work stations.

**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 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 work stations.

**Operating System (OS)** – 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.

**Portable Network Graphics (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.

**Real World Product (RWP)** – The actual Load.In solution as it was intended.

**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 which rents moving vehicles for a Do-It-Yourself Mover to assist them with their move.

**Smartphone** – A device, typically handheld, which can act as both a cellular phone and a computer by running one or more applications through typically 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 work stations.

## References

- 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/>
- Wood, T. (2020, January 6). *Moving Industry Statistics*. MoveBuddha. <https://www.movebuddha.com/blog/moving-industry-statistics/>
- 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 Systems*. 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. (2020, September 20). *Optimizing Three-dimensional Bin Packing through Simulation*. Semantics Scholar. <https://www.semanticscholar.org/paper/OPTIMIZING-THREE-DIMENSIONAL-BIN-PACKING-THROUGH-Dube/bb9986af2f26f7726fceflbc684eac8239c9b853#references>



Economy Moving & Storage, LLC. (2015, January 4). *How to properly pack and load a moving truck- Movers Cincinnati*. YouTube.

<https://www.youtube.com/watch?v=rjmofUZOdwo&feature=youtu.be>

Nat and Friends. (2017, April 18). *Google Earth's Incredible 3D Imagery, Explained*.

YouTube. [https://www.youtube.com/watch?v=suo\\_aUTUpps&feature=youtu.be](https://www.youtube.com/watch?v=suo_aUTUpps&feature=youtu.be)

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>

*The Top 5 Moving Mistakes Across America*. (2019, August 13). Article.

<https://www.article.com/blog/top-5-moving-mistakes/>

US Census Bureau. (2019, October 10). *Historical Households Tables*. The United States

Census Bureau. <https://www.census.gov/data/tables/time-series/demo/families/households.html>