BUILDING A DATABASE FOR A STARTUP: BOLTABOUT

A Senior Project submitted to
the Faculty of California Polytechnic State University,
San Luis Obispo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Industrial Engineering

by
Danielle Tran
ABSTRACT

BUILDING A DATABASE FOR A STARTUP: BOLTABOUT

Danielle Tran

Abstract

BoltAbout is a young start-up based out of San Luis Obispo, CA. Their primary business is renting out bikes via monthly-subscription with the opportunity to lease to own. BoltAbout has seen great success over the past 2 years and is looking to expand. The company currently has 200 bikes, but the demand is far outweighing the supply. The CEO of BoltAbout hopes to grow to grow the company’s inventory to 500 bikes by the end of 2017 and 1500 by the end of the school year. As the company continues to expand, it will need to be able to accurately track their increasing assets and start looking at historical data to forecast future sales. The objective of this project is to create a database that will be able to track their assets, produce accurate forecasts, and monitor customer history.

Utilizing Microsoft Access allows a user to be able to access all information in one convenient location. This means that user interface will be a large factor in determining the success of this project. This database will need to be updated in the future to adjust to future demand so having a solid relational database base to build upon is crucial. Designing these tables and forms requires the information to be easily recorded and accessed.

Ultimately, the database was a success by fulfilling the objectives set out by BoltAbout. It was also successful in being simple, user-friendly and efficient. At the very end of the project, BoltAbout decided to not go forward in uploading their tables to the cloud. Hence naturally, there will be concerns regarding the resilience of this database and the limitation of one user at a time. However, the importance of these two things can be assessed over time and easily implemented by another student or professional after this senior project is completed.
ACKNOWLEDGMENTS

Dr. Tao Yang
James Losack
Tavin Boynton
# TABLE OF CONTENTS

## Contents

I. Introduction ................................................................................................................................. 1  
II. Background .............................................................................................................................. 1  
III. Theory & Design ................................................................................................................... 6  
IV. Methods and Experimentation ............................................................................................ 17  
V. Results and Discussion ......................................................................................................... 19  
VI. Conclusions (or Summary and Conclusions) ...................................................................... 22
LIST OF FIGURES

Figure 1: Visual Representation of Windows Azure ........................................ 2
Figure 2: Types of Demand Classification ....................................................... 4
Figure 3: Components of a Database Management System ............................... 6
Figure 4: SQL Server Functional Diagram ...................................................... 7
Figure 5: Entity-Relationship Diagram ............................................................ 9
Figure 6: Bike Inventory Table – Field Name, Type and Description ................. 9
Figure 7: Battery Inventory Table – Field Name, Type and Description ............ 10
Figure 8: Customer Key Table – Field Name, Type and Description ............... 10
Figure 9: Active Customers Table – Field Name, Type and Description .......... 10
Figure 10: Customer History Table – Field Name, Type and Description ......... 11
Figure 11: Referral Table – Field Name, Type and Description ...................... 11
Figure 12: Main Menu User Interface Design ............................................... 11
Figure 13: Add Customer User Interface ....................................................... 12
Figure 14: Add Bike User Interface ............................................................... 12
Figure 15: Add New Battery User Interface .................................................. 13
Figure 16: Assign Customer to Bike User Interface ........................................ 13
Figure 17: Search and Edit by Bike Information User Interface ....................... 14
Figure 18: Edit Selected Bike Pop-Up Form – Unassigned and Assigned User Interface 14
Figure 19: Search and Edit by Customer Information User Interface ............. 15
Figure 20: Search and Edit by Battery Information User Interface .................. 15
Figure 21: View All Records User Interface ................................................... 16
Figure 22: Insights Form User Interface ....................................................... 17
I. Introduction

This report will describe the process in which the author built a database for the local start-up, BoltAbout.

BoltAbout is a start-up based out of SLO that functions primarily as an electric bike rental company. It provides both subscription based monthly leases with the opportunity to lease to own. It has seen great success over the past 2 years and is looking to expand. The company currently has 200 bikes, but the demand is far outweighing the supply.

BoltAbout does not have a central system that can track an individual bike or battery’s history over time. They also struggle with inventory accountability – they have lost 8 bikes in the past year. They want to be able to know where their assets are at all times. By lessening the unknowns, they are reducing costs that may be lost to events such as lost or stolen bikes or inefficient repair. To accomplish this, the CEO wants more accurate metrics to:

- Track the status of all bikes to ensure accountability, efficient repair and adequate supply
- Produce accurate forecasts to make well-informed decisions
- Monitor customer history to maximize customer retention

There are two main constraints for this project. First, it must be able to fit into the budget of a start-up business. Specifically, the product should cost no more than $500 in the first year and less than $300 per year in upkeep. Second, the database must be able to be updated to adjust to the ebb and flow of pace. Start-ups can experience rapid growth as well as unanticipated downturn at any point in time. The costs and capacity should not be so great that there is unused waste but not so little that the start-up can unexpectedly reach capacity overnight.

There are 2 parts to a database, the front-end and the back-end. The back-end of the database is where the information is stored. The front-end part of the database is how BoltAbout and its employees interacts with the data. This database will be built using Microsoft Access and uploaded to the cloud using Microsoft Azure. All data will be imported from BoltAbout’s pre-existing programs.

II. Background

The CEO of BoltAbout hopes to grow to grow the company’s inventory to 500 bikes by the end of 2017 and 1500 by the end of the school year. As the company continues to expand, it needs more accurate metrics on their bikes. More accurate metrics are needed to track the status of the overall fleet to ensure fast, efficient repair and adequate supply. It can also be used to monitor
customer history to maximize customer retention. In addition to tracking status, accurate data is required in order to make well-informed decisions for future forecasts and decisions.

Current System

Currently, BoltAbout is strictly using Microsoft Excel to record all data on their bikes. It is quick and easy to use but using Excel instead of a real database creates a plethora of complications. First, Excel does not allow for an extensive history to be built upon or forecasted from. By having a system that solely shows the current state of things, they do not allow for a concurrent recording of a bike or customers history. Using the current state also does not allow for company insights. There are no charts or analytical data to draw from and forecast from. Second, excel documents are single-user meaning that only one computer can have the spreadsheet opened at one time. Databases do not have the same limitations and the information within can be accessed by many users at once. Third, Excel is severely limited by memory. Once the file sizes moves upward of 100 MB, there is a huge performance hit. At a maximum, each workbook can hold 256 work sheets that contain 65,000 rows and 256 columns. Alternatively, cloud-based databases can be constantly expanded to fit the needs of the end user. Lastly, using Excel reduces accountability. One person’s mistake in the spreadsheet can wipe out entire records with the click of a button that can’t be recovered.

Literature Review

Introducing the Microsoft Azure Program

The Windows Azure is a program that runs Windows applications and stores data in the cloud.

![Figure 1: Visual Representation of Windows Azure](image-url)
There are 5 parts the make-up Windows Azure:

1. **Compute** – Windows Azure runs applications within a Windows server foundation that are created using the .NET framework. Potential developers can utilize Visual Studio or other development tools to build through C#, Visual Basic, Java or other languages.

2. **Storage** – The storage serves stores blobs, gives queues for communication between the Azure application components and provides a form of tables with simple query language (SQL). Windows Azure applications access the Windows storage service through the REST API.

3. **Fabric Controller** – The Fabric Controllers combines the machines in a single Windows Azure data center into a cohesive, comprehensive whole. This provides a processing power base to build the compute and storage services.

4. **Content Delivery Network** – The Content Delivery Network maintains cached copies of data at sites around the world so that users can have speedier access to that data.

5. **Connect** – Windows Azure Connect provides a firewall for ease of use for organizations to access an on-premise database.

Running Windows Azure and storing data in a cloud-based system has a multitude of benefits. Instead of buying, installing, operating and maintaining its own storage systems, an organization can instead rely on a cloud provider to deliver this service. The organization will only have to pay for the storage and computing they use rather than maintaining a constant, large capacity that must be ready to permit peak loads. The cloud can instead scale better, provide ease and reliability and requires much less hands on administration and maintenance than a traditional database model (Hill).

**Inventory Management and Control**

Finding and implementing a suitable inventory model is one of the most important decisions for an industry. Historically, inventory management has been studied since the second decade of the past century, but this scientific area is still generating a great deal of interest. There is no one standard solution in solving the problem of inventory control. Each company and industry have unique conditions, features and limitations. This article focused on analyzing the possible parameters of preexisting inventory control models.

Every company has stock reserves that are used to carry out the normal activities of the company. By having a well-planned, timely optimal inventory control strategy, resources are used much more efficiently. This will reduce unnecessary costs and keep inventory and stock at optimal profit levels. In controlling inventory, one must ask two questions:
1. How large should an inventory replenishment order be?

2. When should an inventory replenishment order be placed.

In answering the above questions, there are 7 characteristics to be considered: single vs multiple items, time duration, number of stocking points, the nature of product, nature of demand, nature of supply process, and penalty and deficit (Ziukov).

- Single vs multiple items – this characteristic considers whether or not inventory calculations will take into account one item or multiple independent products. This is calculated by observing budget and space constraints and coordinated control/substitutivity.

- Time Duration – For some industries, there is a specific time block where most sales occur. This block is short and at the end of this amount of time the excess stock at the end of the season can’t be used in the next season. If this is the case, a single period model must be used. If implementation is periodically rolled out, a multiple period model must be used. This is the model that BoltAbout is focusing on. For this model, only a small number of future periods are taken to consideration and are recalculated every period start.

- Number of Stocking Points – In some cases, a single stocking point is used in isolation to determine the reorder point. In most other cases, inventories are kept in a variety of locations and may even be linked to the demand at a different location. Incorporating a reordering system that utilizes linear programming is beneficial in determining these specific reorder points.

- The Nature of the Product – The product type helps identify and determine product characteristics. Some things to take into consideration is the reparable, consumability, recoverability and perishability. Products naturally deteriorate in storage and therefore must be seriously considered in the inventory policy. Demand can be classified by the Figure 2 below.

Figure 2: Types of Demand Classification
• Nature of Supply Process – This characteristic includes any constraints that are imposed on the inbound process of the supply chain system. Things to consider include reorder lead times and minimum order size.

• Penalty and Deficit – Lack of stock at the right time can lead to losses due to downtime, unevenness of production. This characteristic is the consequence for a miscalculation of those above.

It is a difficult task to quantify realistic values for the inventory model parameters outlined above. In many cases, these parameters are hazy and unknown and add a degree of uncertainty to the result. To mitigate this, one must incorporate vague models called fuzzy models. Fuzzy set theory allows the user to deal with the uncertainty in a defined manner. In the real world, there are two types of uncertainty: environmental and system uncertainty. Environmental uncertainty involves issues outside of the production process such as that of supply and demand. System uncertainty includes those included within the production process, such as operation yield, production lead, and quality uncertainty.

The modern state of inventory management deals with the inclusion of new models that take into account uncertainty. BoltAbout will need to give additional thought to the way they manage their inventory. Currently, they have a very ambitious plan of expansion, 300% within a year. This goal is not backed by any quantifiable data. From research in this literature review, the author thinks that BoltAbout should go about defining parameters within an inventory model to determine the optimal reorder point for their bikes and batteries.

The apparatus, method and system for distributed and automatic inventory, status and database creation discusses the need for an apparatus, method, and system in order to provide concise inventory management and control. Each apparatus, method and system should be able to inventory and track each SKU, status, location and history in order for BoltAbout to have professional, accurate, and detailed information to make intelligent business decisions (Ziukov).

Relational Database Design

A database management system, also known as a DBMS, is software that stores data and provides the necessary software packages for manipulating the stored data. A database is a collection of data that is stored and managed by the DBMS. See Figure 3 to see the components of a database management system.

Specifically, a relational database is a database that contains tables to store sets of data. Relationships can be drawn between these different sets of data within the database. See Figure 5 to see the Entity-Relationship diagram that defines the RDBMS for the BoltAbout database.
The primary goal of a relational database is that instead of storing data within a tree-like or directed graph data structure, they are instead stored in their natural state in tables. These will instead have variables within each table that link to one another through relationships. At least one column of the table requires a reference data type, the primary key, which will be used to reference another variable in a different table. Through these relationships, instead of compiling too much information in the rows and columns of a chart, the data is more efficiently and intelligently handled by the user (Goldberg & Jirak).

III. Theory & Design

Theory

BoltAbout would like the author to create a functional database that will track all bikes, maintenance history and customer history and output executive reports to monitor costs, trends and current fleet status. It will need to be composed of both a back-end and a front-end. The details of the theory and design are described in the following.
Data and applications are stored on a hard drive or a series of hard drives or other media, such as a cloud. All of this storage is managed by the operating system. The operating system, which is Windows in this diagram, handles all of the low-level tasks associated with running applications as well as managing storage. SQL Server runs on top of Windows. SQL Server is actually dozens of modules, which are separate pieces of software. Each module is called when needed. This diagram has been simplified to show only the query processor and security. The query processor translates SQL into a form of code the other server modules can use. All communications with the server go through the security layer. The security layer is a blending of operating system functions and functions specific to SQL Server. SQL is the communications protocol between the SQL Server and the outside world. Most communications with SQL Server are in Standard Query Language (SQL). On the right, there is a box called the SQL Server Management Studio. SQL Server Management Studio gives us a single location to manage our SQL Servers and develop and test our databases and SQL code. Before one can manage servers or run SQL code, they must set up a connection to each server they want to work with. The servers can be anywhere. The only requirements are a network connection and security clearance.
For BoltAbout, the back-end portion of this database needs to be simple, efficient, economical and agile. As a start-up, it needs to have the ability to expand and shrink drastically depending on quarter performance. This project will be utilizing Database-as-a-Service which is a software that enables users to provision, manage, consume, configure and operate database software. It is operable by using computing primitives to interact with software without needing to know its exact implementations. Database-as-a-Service is adaptable - perfect for applications that require quick changes in scale, such as for start-ups. They are economical, only a few dollars per month to maintain a subscription. Database-as-a-Service is also agile and works well with dynamic applications to increase agility, organization and accessible, useable data.

Specifically, this project will be utilizing Microsoft Azure. Microsoft Azure will act as the facilitator that connects certified databases such as Oracle and MySQL to the developer and end user. This simplifies the provision and management of databases to offer its users self-service, managed databases for database platforms. To summarize, Microsoft Azure is where the database will be hosted and uploaded to.

The database that will be uploaded is linked through Microsoft Access. Offline, the author will create the database and link the tables as well as import all data that BoltAbout currently has. This will then be uploaded and be brought online through Microsoft Azure.

**Front-End**

The Front-End of the database consists of the client-side interaction. The primary objective of the front-end is to provide an interface to fetch, store and display the data stored within the database. This is required because users should not be able to directly manipulate data without accountability. This can be done by adding format filters that run when the user attempts to store or update data. When interacting with the front-end the data should be pleasant and easy to look at. For BoltAbout specifically, this will need to be simple and easy to use and gain insight from. Employees who are not naturally tech-savvy will need to be able to navigate through front-end and be able to gain insight from their interaction.

**Design**

The software that the author chose to use for this project was Microsoft Access due to the extensive knowledge foundation the author have from the IME 312 course, Database Management Systems. This course focused heavily on Microsoft Access as it is a basic, easy to use, entry-level software that can fulfill all the requirements set about by BoltAbout. It also has the capability to be converted to another platform such as SQL Server and is compatible with software such as Microsoft Azure and Tableau. Utilizing this software is ideal due to the knowledge gleaned from the Cal Poly IME curriculum instead of purchasing and learning a new software package that may not be as easily manipulated or updated by BoltAbout in the future.
The design effort was a collaboration between the author and BoltAbout management who sought to fulfill the objectives set at the beginning of this project.

The first step was to design the appropriate tables required to store all the required data that BoltAbout wants to keep track of. Functional tables will later be created to perform functions for executing searches. The below figures show the Entity-Relationship diagram for the database and the Field Name, Variable Type and Description for each table.

![Entity-Relationship Diagram](image)

**Figure 5: Entity-Relationship Diagram**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description (Optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>AutoNumber</td>
<td>Primary Key</td>
</tr>
<tr>
<td>bikeserial</td>
<td>Short Text</td>
<td>Unique identifier for bike</td>
</tr>
<tr>
<td>bikelabel</td>
<td>Short Text</td>
<td>Accompanying identifier for bike</td>
</tr>
<tr>
<td>color</td>
<td>Short Text</td>
<td>Color of the bike</td>
</tr>
<tr>
<td>date_rec</td>
<td>Date/Time</td>
<td>Date that the bike was received in inventory</td>
</tr>
<tr>
<td>bikemansufacturer</td>
<td>Short Text</td>
<td>Bike manufacturer name</td>
</tr>
<tr>
<td>keynum</td>
<td>Short Text</td>
<td>Number corresponding to unlock key</td>
</tr>
<tr>
<td>model</td>
<td>Short Text</td>
<td>Bike model</td>
</tr>
<tr>
<td>condition</td>
<td>Short Text</td>
<td>Condition the bike is in</td>
</tr>
</tbody>
</table>

**Figure 6: Bike Inventory Table – Field Name, Type and Description**
**Figure 7: Battery Inventory Table – Field Name, Type and Description**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description (Optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>AutoNumber</td>
<td>Primary Key</td>
</tr>
<tr>
<td>batterylabel</td>
<td>Short Text</td>
<td>Accompanying identifier for battery</td>
</tr>
<tr>
<td>batterymanufacturer</td>
<td>Short Text</td>
<td>Battery manufacturer name</td>
</tr>
<tr>
<td>style</td>
<td>Short Text</td>
<td>Style of frame the battery fits</td>
</tr>
<tr>
<td>voltage</td>
<td>Number</td>
<td>Voltage of battery (V)</td>
</tr>
<tr>
<td>amperage</td>
<td>Number</td>
<td>Amperage of battery (Ah)</td>
</tr>
<tr>
<td>date_rec</td>
<td>Date/Time</td>
<td>Date that the battery was received in inventory</td>
</tr>
<tr>
<td>condition</td>
<td>Short Text</td>
<td>Condition the battery is in</td>
</tr>
<tr>
<td>batteryserial</td>
<td>Short Text</td>
<td>Unique identifier for battery</td>
</tr>
</tbody>
</table>

**Figure 8: Customer Key Table – Field Name, Type and Description**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description (Optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>custkeyid</td>
<td>AutoNumber</td>
<td>Primary Key</td>
</tr>
<tr>
<td>fname</td>
<td>Short Text</td>
<td>Customer first name</td>
</tr>
<tr>
<td>lname</td>
<td>Short Text</td>
<td>Customer last name</td>
</tr>
<tr>
<td>email</td>
<td>Short Text</td>
<td>Customer e-mail</td>
</tr>
<tr>
<td>birthday</td>
<td>Date/Time</td>
<td>Customer birthday</td>
</tr>
<tr>
<td>license_no</td>
<td>Short Text</td>
<td>Customer drivers license number</td>
</tr>
<tr>
<td>address</td>
<td>Short Text</td>
<td>Customer address</td>
</tr>
<tr>
<td>cell</td>
<td>Short Text</td>
<td>Customer phone number</td>
</tr>
<tr>
<td>custid</td>
<td>Short Text</td>
<td>Unique identifier for customer</td>
</tr>
</tbody>
</table>

**Figure 9: Active Customers Table – Field Name, Type and Description**

<table>
<thead>
<tr>
<th>Field Name</th>
<th>Data Type</th>
<th>Description (Optional)</th>
</tr>
</thead>
<tbody>
<tr>
<td>actveid</td>
<td>AutoNumber</td>
<td>Primary Key</td>
</tr>
<tr>
<td>bikeserial</td>
<td>Short Text</td>
<td>Unique identifier for bike</td>
</tr>
<tr>
<td>batteryserial</td>
<td>Short Text</td>
<td>Unique identifier for battery</td>
</tr>
<tr>
<td>bikelabel</td>
<td>Short Text</td>
<td>Accompanying identifier for bike</td>
</tr>
<tr>
<td>sub_start</td>
<td>Date/Time</td>
<td>Subscription start date</td>
</tr>
<tr>
<td>batterylabel</td>
<td>Short Text</td>
<td>Accompanying identifier for battery</td>
</tr>
<tr>
<td>sub_end</td>
<td>Date/Time</td>
<td>Subscription end date</td>
</tr>
<tr>
<td>package</td>
<td>Short Text</td>
<td>Type of package purchased</td>
</tr>
<tr>
<td>lockkeynum</td>
<td>Short Text</td>
<td>Bike key identifier</td>
</tr>
<tr>
<td>custid</td>
<td>Short Text</td>
<td>Unique identifier for customer</td>
</tr>
</tbody>
</table>
The final database design was able to fulfill the necessary objectives for project as well as left room for potential expansion. This database will be functional in its current state and will be able to be expanded to meet the requirements of the upcoming bike inventory expansion. The following figures show the final design of the database user interfaces used to perform everyday tasks.
The Main Form design allows for the categorization of form design. Under Manage, the user can either add a new record for a bike, battery or customer; or assign a bike and battery to a customer. Under Search, the user can search based either bike, customer, or battery information; or be able to view all records. The Insights category will lead the user to view the up-to-date insights pulled from current and historical data.

*Manage*

![Add New Customer User Interface](image1.png)

*Figure 13: Add Customer User Interface*

![Add New Bike User Interface](image2.png)

*Figure 14: Add Bike User Interface*
Within the Manage forms, the User can fill out the listed information required to add a new record to the system. In the Assign Customer to Bike User Interface, the user can search by any of the listed identifying factors, click search, and have the subform populated with records that fulfill the correct criteria. The user should then select the appropriate customer, bike and battery to populate the form portion shown at the bottom of the right panel. Finally, click “Assign Bike” to add the record which will be sent to the active customers table.
Search

Figure 17: Search and Edit by Bike Information User Interface

Figure 18: Edit Selected Bike Pop-Up Form – Unassigned and Assigned User Interface
Figure 19: Search and Edit by Customer Information User Interface

Figure 20: Search and Edit by Battery Information User Interface
The above Search and Edit Figures show the different criteria that can be used to filter through Active Customers and Inventory for the item type specified. The subform is filtered in a similar way, the user simply fills out the text boxes with the information they want to filter by then press search. Both subforms will refresh and you can view the remaining records. To edit the record, the user will click on the row in the subform that contains the record and click “Edit Information”. To view the historical record for the item, instead click “View Battery History”.

In the View All Records User Interface, the user can select any of the shown buttons to show all records for any of the 6 tables we created. This helped verify data was correctly appended, updated and deleted during the testing phase. It will also help the end user visualize their data in a compiled manner.

**Insights**

Under the lone Insights form, there are 4 visual aids that will help the user gain insight from the data. The first aid is an output of the Total Number of Ready Bikes and the Total Number of Broken Bikes. This will help the end user known instantaneously how many bikes need attention and how many are prepared to be assigned to customers. With this knowledge, more efficient repair can be done to balance supply and demand. The Average Subscription Length Purchased
per Month shows visually how many customers are added per month. Using this data, the CEO can encourage a sales push to increase the number of new customers or be able to forecast future demand. Similar actions can be taken after viewing the Subscriptions Purchased per Month graph. The Package Breakdown shows the current popularity of each package. From this data, the CEO can push to raise investment money in adding more bikes of one model to their fleet over another.

![Insights](image)

**Figure 22: Insights Form User Interface**

### IV. Methods and Experimentation

There are two types of testing that was done for my project: User-Interface Testing and Data Testing.

*User-Interface Testing*
User-interface testing primarily consists of all the testable items that can be viewed and interacted with by the user such as Forms, Presentations, Graphs and Reports. To test the User-Interface, the author inputted test data to simulate the adding of new bikes, batteries and customers. the author also simulated the edit and deletion functions by simulating tasks such as bike assignment and subscription cancellation.

Data Testing

Data Testing primarily targets all of the testable items that are generally hidden from the user. This type of testing involves validating the schema, database tables, columns keys and indexes, stored procedures, triggers, database server validations and data duplication validations. There are 3 types of data testing: structural, functional and non-functional.

Structural Data Testing – Normally involves the validation of elements within the data tables that are used primarily for storage of data and are not allowed to be directly manipulated by the end users. Structural data testing is tested in the following ways:

i. Table and Column Testing –
   a. Checked whether the mapping of the database fields and columns in the back end was compatible with those mappings in the front end by manual data entry. The data was then manually checked to verify appropriate mapping.
   b. The length, data type and naming convention of the database fields and columns were validated to fit the requirements set by BoltAbout.
   c. Removed all unused or unmapped tables and columns.
   d. Verified that the database fields allow the user to provide desired user inputs as required by the business requirements set by BoltAbout.
   e. Confirmed that the primary key and foreign key constraints have been created on the required tables.
   f. Verified that the data types of the primary key and corresponding foreign keys are standardized.
   g. Verified that the required naming conventions have been followed for all keys.

ii. Procedure Testing (Microsoft Azure)
   a. Verified that the appropriate exception and error handling for all stored procedures for all modules were tested – Ensured that no error messages appeared when performing all functions of the database.
b. Tested all conditions and loops by applying the required input data to the application.

c. Confirmed that the manual execution of all procedures provides the end user with the required result. This includes inputting, updating and deleting.

d. Validated allowance of all Null conditions at the database level.

e. Verified that all stored procedures and functions can still be successfully executed when the database tables are blank.

f. Verified that all triggers fulfill the necessary conditions and updates the data correctly once executed – ex. Opening “Insights” panel triggers automatic form updating with the correct data.

g. Validated the overall integration of the stored procedure modules to fulfill the requirements of the database as set by BoltAbout.

Functional Data Testing adheres to the requirements needed to ensure that the transactions and operations performed by the end users are consistent with the requirements. Functional Data Testing is done by:

i. Checking Data Integrity and Consistency

   a. Confirmed that data is logically well organized and stored in tables as per the business requirements.

   b. Removed all unnecessary data present in the application under test.

   c. Verified that all data is stored as per the requirement with respect to the data uploaded, edited or deleted in the user interface.

Non-Functional Testing is synonymous with Performance Testing and has 2 primary focuses: Risk quantification and setting a minimum system equipment requirement. For my project, the author decided not to run Non-Functional testing (Load and Stress testing) because of the small size of the database. There is not a need to stress Microsoft Access until system failure in order to identify the breakdown points of the system.

V. Results and Discussion

The result for this project exceeded the initial expectations. The final design, as shown in the Entity-Relationship diagram, was carefully thought out and mapped out to fulfill the requirements set out by BoltAbout. The final structure is a Relational Database that is designed
in such a way that is straightforward and allows for future expansion, should BoltAbout desire. Potential areas of improvement include new products and models of bikes or batteries. Due to the lack of row categories, if BoltAbout would like to add two more models of bikes, they can simply input it into the text field and it will be able to be sorted and filtered by. Overall, the quality, cost and productivity estimates were on track by the end of the project and the overall design was successful.

The primary objective for this project was to incorporate the ability to know where all assets are at all times. BoltAbout wanted to lessen the unknowns, such as stolen bikes in order to reduce waste due to lost inventory. By introducing a database in which the tables “talk” to one another, there will never be a record lost or accidentally deleted. BoltAbout also wanted to be able to have an accurate screenshot in time of the number of available bikes and bikes that need repair. The accountability of data in my database means that the user will always know the current # of fixes and will be able to prioritize fixing to balance supply and demand. Thirdly, BoltAbout wanted the ability to gain historical insight from the data so that the user can make informed decisions. The Insights User Interface is a form that is updated upon opening and gives 4 visual aids for the user to choose from.

The results were limited at the very end to a local database without the cloud. At the end of the project, when given the database size and limitations, BoltAbout decided against uploading their database to the cloud. Since this change did not require any action until the very end, this did not negatively affect my design. The only difference in the actions that would have been taken is the uploading of the tables into Microsoft Azure via an ODBC Connection. This limits the size of the database to the standard 2G size. Since the current size of the database with all the information uploaded is < 4,000 KB, there are 500x more space remaining. This will also limit the number of users that can access the database at one time. However, the CEO emphasized that since there are only ~7 employees, it is highly doubtful that the users will conflict.

Looking forward, the author believes that it will be a good idea for BoltAbout to move to the cloud within the next 6 months. Since every Industrial Engineering major at Cal Poly takes IME 312 and learns how to use Microsoft Access, the author does not believe that BoltAbout will struggle with finding another individual to do this for them. They will also be able to update and edit the database in the same way as the company expands. Some problems may come up if the database has some sort of system error. Since they do not have anyone in-house providing IT, they may be stuck until they can call someone in to fix it.

**Economic Impact**

**Quantitative**

There is no quantitative cost for the running of the current system. Since the software that is being used, Microsoft Excel, comes free for most pre-built computers, BoltAbout is not currently
sinking additional costs into their data management system. The main source of the quantitative cost comes from the lack of accountability for lost or damaged items. By implementing my database, BoltAbout can potentially create cost savings from locating and fixing this equipment. They can also eliminate costs from the time lost to finding and repairing.

**Qualitative**

Most of the costs involved with the current system involve risk, efficiency and productivity. The current system in use is not very efficient. Since it is a public excel worksheet, only one person can edit the worksheet at one time. If the spreadsheet is open by another user, then all other employees can only open a read only version of the document. All read-only edits will not be saved unless a copy of the spreadsheet is made which creates confusion as to which is the most up to date. This contributes to a limit of productivity. Additionally, there is no system in place to view historical data. Once the cell is changed, it is changed permanently. If a mistake was made, then the old data cannot be retrieved. As their customer base increases, bikes and batteries can potentially be lost or damaged, and keeping track of this equipment and customers is essential for accountability purposes. Hence, employees will have to cease the task they are working on and spend hours tracking and fixing the incorrectly recorded or missing information. The temporary nature of the data also creates limitations on what insights can be pulled from the data. The opportunity cost is lost through decisions that could potentially be made and profited from through the visualization and interpretation of this data.

**Ethical, Societal and Environmental Impact**

I have identified 3 major changes in terms of organizational impact: changes in structure, changes in culture and changes in knowledge.

*Changes in Structure and Nature of Work*

The utilization of my project will affect the way that the company, Bolt About, is structured. By introducing a new system that ties in all record-keeping aspects of the company, the author is eliminating the usage of the individual software that is currently being implemented. The company will have to adapt to the new way of record-keeping. The structure will also prevent data corruption and accidental loss of data history.

*Changes in Knowledge*

The introduction of new technology changes the knowledge resources of an organization. The accessible storage of information can now be shared and edited safely across the company. However, this new technology will require new skills to be learned in order to utilize it. As an effect, old knowledge sets will be eliminated and no longer used.

*Changes in Culture*
Before introducing new technology, the author will also need to consider how the way that people use technology will affect the way things are currently done. Currently, employees have access to edit and view the data points themselves, but they aren’t organized into viewable insights. Depending on what access the CEO wants to give to his employees, they may be given additional transparency or have their visibility restricted.

**VI. Conclusions** (or **Summary and Conclusions**)

The CEO of BoltAbout hopes to grow the company’s inventory to 500 bikes by the end of 2017 and 1500 by the end of the school year. As the company continues to expand, it needs more accurate metrics on their bikes. However, BoltAbout does not have a central system that can track the location of all assets at all times. This lack of inventory accountability leads to money lost to events such as lost or stolen bikes or inefficient repair. The objective of this project then was to create a database that will be able to track the status of all bikes, provide accurate forecasts and monitor customer history.

BoltAbout’s current state is therefore too small and inefficient to keep up with the company’s growth. The current method used to process orders, add and manage customers and equipment is not appropriate and does not provide the functions needed by BoltAbout. Data can easily be misplaced or deleted which result in lost equipment, lack of accountability, and erroneous order information.

This projected addressed the above objectives and produced the results below:

- Created a relational database that ensured that the status of all bikes, batteries and customers could be accounted for at all times.

- Created a historical equipment log in order to develop a maintenance and rental history for each item. This data was used to produce accurate forecasts so that well-informed decisions could be made.

- Created a log for customer history to keep track of customer trends in order to maximize customer retention.

- This database stayed within the 2 main constraints of the project. First, this product had no cost due to the lack of cloud usage. If the cloud was to be utilized, it would have cost $5/month. Second, the design of this database allows for the adjustment and updating by another individual, ideally another IME 312 student.

Through this senior project, the author learned the importance of planning and utilizing resources when they are available. While the author had previously believed they had extensive knowledge of Microsoft Access, there were countless hours sunk unnecessarily when the author became
stuck. If the author had either planned for the time lost or utilized my resources – my technical advisor, Tao Yang and his teaching assistant, James Losack – the author would have been able to overcome these obstacles faster. The final product was well outside of my capabilities at the beginning of this project. Constant learning and brainstorming with the academic advisor, research assignment, client and other resources is absolutely necessary for the completion of this senior project. Looking forward, the author hopes to apply this knowledge to my future academic and professional career.

Figure 23: Confirmation of BoltAbout Satisfaction
REFERENCES


