Title Search

Cycle 1 Report

Brandon Toups (bmt0015)

Justin Greenfield (jrg0041)

Mina Narayanan (mjn0010)

AJ Hancher (ajh0067)

Jay Patel (jpp0016)

Sponsored by

Trey Tinnell, IBM

Verified by \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ on \_\_\_/\_\_\_/\_\_\_

Signature \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Auburn, Alabama

February 4, 2018

Comp 4710 Senior Design Project

**Table of Contents**

[**System Metaphor**](#_yzmxasiv13oj) **3**

[**Introduction**](#_ibxne5cz8jdy) **4**

[**User Stories**](#_q6wrc2q6bey) **4**

[User Story 1](#_set2homm0n27) 4

[User Story 2](#_xm19c4h0l5hy) 5

[User Story 3](#_26j4bk415o8a) 5

[User Story 4](#_rijf8lh8ie1e) 6

[User Story 5](#_tgvy4a6lheqg) 6

[User Story 6](#_mouonl8fgmak) 7

[User Story 7](#_3eodbsweyib6) 7

[User Story 8](#_npjizesyfiu1) 7

[User Story 9](#_4wc2hy309no9) 8

[User Story 10](#_b1pzeumewk5g) 8

[User Story 11](#_apjumgwwquyp) 9

[User Story 12](#_t1xuoeu6qos3) 9

[User Story 13](#_z3uhgmgfx1v1) 10

[User Story 14](#_2nazsauqsl7) 10

[**Process Description**](#_5tfuwd9jxhof) **10**

[**Design Documentation**](#_gl17l4bo88q6) **11**

[Languages](#_bh135vssbmk5) 11

[Tools](#_xozql4a8ygad) 11

[UI](#_l1bhwt8ch4ls) 12

[Architecture](#_h1gtrd26uciw) 12

[Structure](#_kkojiadup5k3) 12

[Sequence Diagrams  
TitleSearchManager Sequence Diagrams](#_cbccld9rlmfh) 14

[WarrantyDeedManager and TitleSearcherComManager Sequence Diagrams.](#_7abeft84ajk) 15

[**Management Plan**](#_gws02jvkhyuu) **18**

[Task Assignment](#_9knt8eymyzen) 18

[Development Schedule](#_g2d9tlv5kra9) 19

[Planned Code / Feature Freeze](#_u7sptldf6ao6) 19

[**Risk Mitigation**](#_kr79634w83e7) **19**

[**Test Plan**](#_vc43eghnad13) **19**

[**Test Procedure**](#_xu580zpdkvhn) **19**

[**Lessons Learned**](#_z9zt0s5tc3q5) **20**

[**Test Documentation**](#_484jyl9jgkfj) **20**

[**Memoranda**](#_z84yafar171x) **20**

[Meeting Minutes](#_fzl7tzkcv8gi) 20

[Email Correspondence](#_fhna6d8fas46) 22

[Whiteboard Pictures](#_kqizvh8ilbdl) 33

[**Version Description**](#_qu6ue050gqsj) **34**

# System Metaphor

Author: Mina Narayanan

People seeking to purchase real estate typically pay a lawyer to perform a “title search” to verify that none of the previous owners of the property over the past 40+ years still have any outstanding loans or liens on the property that is for sale. If any existing liens or loans on the property aren’t resolved before purchasing the property, the potential buyer could end up losing the property.

The lawyer is responsible for searching through physical copies of Warranty Deeds (which show ownership) at a local Registrar’s office or sifting through online records of deeds to ensure that there are no outstanding claims against a property. This process is expensive in terms of time and money. The aim of our software is to use artificial intelligence to automate this search, allowing people to bypass lawyers in lieu of quickly finding out property information for themselves. By entering information into the application, such as a warranty deed book & page #, a user can easily navigate through the title deeds on a property from the last 40+ years and extract information necessary to make an informed purchase.

In the first phase, the system should be able to generate a list of the names of all the previous owners of a particular piece of property.

# Introduction

Cycle Intent

Author: Mina Narayanan

The intent of this cycle is to implement the TitleSearch project using an Object-Oriented approach. We will create 7 classes (WarrantyDeed, WarrantyDeedManager, TrustDeed, TrustDeedManager, TitleSearch, TitleSearchManager, and DatabaseManager) that will interact with the BeautifulSoup and NLU APIs and pass the Warranty Deed and Trust Deed documents as an object while carrying out a specific task. These tasks are outlined in our User Stories and Class Sequence diagrams.

Previous Development

Author: Mina Narayanan

We have developed a front-end TitleSearcher UI, rudimentary back-end functionality, a prototype for our MongoDB database, and BeautifulSoup and Natural Language Understanding APIs. We also have an end-to-end prototype running, where functionality that we have yet to implement has been replaced with code stubs.

Future Work

Author: Mina Narayanan

We plan to train Watson Knowledge Studio on approximately 200 Warranty Deed documents during Cycle 2. We will fine-tune our system architecture and expand it to extract and analyze Trust Deeds. We plan to finalize how we uniquely identify documents and how we differentiate pre-1993 and post-1993 property document extraction.

# User Stories

## User Story 1

Author: Mina Narayanan

**User Story:** As a user, I want to enter a Warranty Deed book number and a page number into the application and receive a PDF version of the desired document.

**Description:** After entering a Warranty Deed Book number and a Page number, the user is directed to a page that either contains an image of the desired document, or a message indicating that no documents matched the search criteria.

**Task:** Develop a UI and an API that interfaces with the UI and TitleSearcher.com to extract images associated with a given Warranty Deed Book number and a Page number

**Summary:** The developers must gain familiarity with how searches are conducted on TitleSearcher.com to successfully integrate the website with the UI and API.

**Planned Hours:** 5

**Actual Hours:** N/A

**Coder(s):** Mina Narayanan, Jay Patel, AJ Hancher

**Tester(s):** Mina Narayanan, Jay Patel, AJ Hancher

**Status:** Completed

## User Story 2

Author: Mina Narayanan

**User Story:** As a user, I want to convert the PDF image of the property document into a text document.

**Description:** After receiving a PDF image of a document that matches the search criteria, the user is able to select whether he or she wishes to convert it to a Word or text document.

**Task:** Develop a IBM Watson Discovery Optical Character Recognition (OCR) API to convert the image to a Word or text document

**Summary:** The developers must create a back-end Java server that can pass images to IBM’s Watson Discovery service to be converted into text data

**Planned Hours:** 2

**Actual Hours:** N/A

**Coder(s):** Jay Patel, AJ Hancher, Brandon

**Tester(s):** Jay Patel, AJ Hancher, Brandon

**Status:** In Progress

## 

## User Story 3

Author: Jay Patel

**User Story:** As a user, I want an UI where I can input the warranty book number and page number to get a report on the house with the necessary details.

**Description:** The web application needs a UI where the user can visually search for the warranty deed book # and page # to find the specific warranty deed.

**Task:** Create a user interface that provides two input fields where the user can search for a specific warranty book number and page number.

**Summary:** Create a frontend user interface using HTML, CSS, and JSP to communicate the query of the user to the backend.

**Planned Hours:** 3

**Actual Hours:** 3

**Coder(s):** Jay Patel

**Tester(s):** Team

**Status:** Completed

## User Story 4

Author: Justin Greenfield

**User Story:** As a user I want to call a service that analyzes text documents and picks out valuable data

**Description:** After calling the IBM Watson Natural Language Understanding (NLU) API and obtaining unique API service credentials [api key, url] the user can use the NLU service to analyze text documents and input text for certain values of sentiment, concepts, categories, entities, and keywords along with identifying subject targets and detecting sentence context

**Task:** Call the NLU API to analyze webpage text and returns custom developer-specified values

**Summary:** The developers must use Watson’s NLU service to decipher specific text values and elements from Warranty Deed text documents.

**Planned Hours:** 3

**Planned Hours this Cycle:** N/A

**Actual Hours:** 2

**Hours this Cycle:** N/A

**Coder(s):** Justin Greenfield

**Tester:**

**Status:** in progress

## User Story 5

Author: Justin Greenfield

**User Story:** As a user I want to call a service that analyzes Warranty Deed documents and returns the name of the Grantor (seller) and Grantee (buyer).

**Description:** NLU can be used via API call to analyze Warranty Deed documents and obtain valuable programmer-specified information such as the Grantors and Grantees of various real estate properties

**Task:** Develop an app that calls the NLU API to analyze Warranty Deeds and return the name of the Grantor (seller) and Grantee (buyer)

**Summary:** The developers must use Watson’s NLU to decipher specific text values from Warranty Deeds

**Planned Hours:** 5

**Planned Hours this Cycle:** N/A

**Actual Hours:** N/A

**Hours this Cycle:** N/A

**Coder(s):** Justin Greenfield, Brandon Toups

**Tester:**

**Status:** in progress

## User Story 6

Author: Jay Patel

**User Story:** As a user, I want to be able to search any term in a scanned document that has been processed by the IBM Discovery service.

**Description:** The IBM Discovery service provides an api to query the scanned documents processed using OCR.

**Task:** Develop a page that allows the user to type in a query and the api endpoint return a QueryResponse object that contains the response from the IBM Discovery service.

**Summary:** Create a page that allows you to query terms in the warranty deeds.

**Planned Hours:** 5

**Actual Hours:** N/A

**Coder(s):** Jay Patel

**Tester(s):** Team

**Status:** In Progress

## User Story 7

Author: AJ Hancher

**User Story:** As a user, I want to be able to store the deeds in a functional database.

**Description:** Creating a database that will be used to store valuable information that is pulled out of the documents. The database will also store the PDF image of the Warranty & Trust Deeds. The database has to be organized in a way that will store deeds on the same property together as well as being able to store who the grantors and grantees are.

**Task:** Develop a database that can store the deed and the valuable information that is required for a search.

**Summary:** The developers must use mongoDB to be able to create a database.

**Planned Hours:** 15

**Actual Hours:**

**Coder(s):** Alan Hancher

**Tester(s):**

**Status:** in progress

## User Story 8

Author: Brandon Toups

**User Story:** As a user, I want to be able to have Watson analyze types and subtypes relationships by

**Description:** I want Watson to be able to parse types (people, for example) and subtypes (Grantors/Grantees are subsets of people, for example).

**Task:** While annotating types and subtypes, create relationships between entities that help reinforce the knowledge base that Watson can analyze.

**Summary:** By recognizing the way the entities work together, we can help Watson gather information that is domain specific, making it stronger at analysis for deeds.

**Planned Hours:** 20

**Actual Hours:** N/A

**Coder(s):** Brandon

**Tester(s):** Team

**Status:** In progress

## User Story 9

Author: Brandon Toups

**User Story:** As a user, I want Watson to learn how to automatically recognize and extract Grantors (sellers) and Grantees (buyers) information from Warranty Deeds.

**Description:** We want to use IBM’s Watson Knowledge Studio to train Watson to build a machine learning natural language processing (NLP) model that understands how to read a Warranty Deed. We do this by building a knowledge base, known as Ground Truth, and annotating hundreds of sample Warranty Deed documents. Watson NLU and Watson Discovery are able to leverage this machine-learning model to understand and extract relevant information from the real estate documents.

**Task:** Have developers annotate the various documents to make Watson able to understand language from new inputs.

**Summary:** Create annotation tasks via Watson online portal, and adjudicate any contradictory annotations. Push clean annotations to the Ground Truth.

**Planned Hours:** 5

**Actual Hours:** N/A

**Coder(s):** Brandon

**Tester(s):** Team

**Status:** In progress

## User Story 10

Author: Brandon Toups

**User Story:** As a user, I want Watson to understand domain specific knowledge.

**Description:** Watson understands certain entities and relationships by default, but creating these entities and relations requires domain-specific knowledge.

**Task:** Developers need to develop a custom type-system for both entities and relationships, so that annotating and later parsing will be possible.

**Summary:** This aids in Watson understanding domain-specific knowledge, as opposed to just being able to analyze entities and relationships. Custom type systems allow the paring of warranty deeds and trust deeds.

**Planned Hours:** 20+

**Actual Hours:** 1

**Coder(s):** Brandon

**Tester(s):** Team

**Status:** Pending

## User Story 11

Author: Jay Patel

**User Story:** As a user, I want to be able to generate report of the history of a house using warranty deeds.

**Description:** After Watson NLU has parsed the Warranty Deed text, it needs to be stored in object member variables and saved to the database in order to be fetched and presented as a report on the UI

**Task:** Developers need to interface the NLU API and the TitleSearchManager classes in order to present grantor, grantee, warranty deed txt, book number, and page number

**Summary:** Connecting the NLU API and TitleSearchManager classes will allow the parsed Warranty Deed elements to be stored in a warranty deed object and presented on the User Interface

**Planned Hours:** 4

**Actual Hours:** 2

**Coder(s):** Jay, Justin

**Tester(s):** Team

**Status:** Pending

## User Story 12

Author: Jay Patel, Justin Greenfield

**User Story:** As a user, I want validation of NLU entity extraction by means of UI input boxes.

**Description:** After the NLU document entities have been successfully extracted and presented on the UI, the user needs to be able to edit the values of entities in order to verify correctness

**Task:** Develop a way to edit/verify the analyzed entity text once it has been presented on the user interface

**Summary:** Verifying the entities will allow the TitleSearch application to provide more correct, concise grantor and grantee names

**Planned Hours:** 4

**Actual Hours:** 1

**Coder(s):** Jay, Justin, Brandon, Mina, AJ

**Tester(s):** Team

**Status:** Pending

## User Story 13

Authors: Jay Patel, Brandon Toups, Justin Greenfield

**User Story:** As a user, I want to view house history step by step with validation using UI

**Description:** The user needs to be able to step through the property warranty deed information on the user interface while viewing the respective entities and property description.

**Task:** Develop a way to step through information on the UI (i.e. carousel?)

**Summary:** Stepping through the Warranty Deed information will allow the TitleSearch application to provide more correct, concise grantor and grantee names

**Planned Hours:** 3

**Actual Hours:** 1

**Coder(s):** Brandon, Jay, Justin, Mina, AJ

**Tester(s):** Team

**Status:** Pending

## User Story 14

Author: AJ Hancher

**User Story:** As a User, I want to be able to retrieve a Deed from the database.

**Description:** After a user types in what deed they want to view on the UI, the database class manager takes this information in and retrieves the correct document from the database to be sent back to the user interface.

**Task:** Develop the database manager class that contains a get function to retrieve that correct document.

**Summary:** The Database class manager will contain a get function that takes in the warranty deed object and returns the correct document in the database.

**Planned Hours: 8**

**Actual Hours: N/A**

**Coder(s):** AJ

**Tester(s):**

**Status:** In Progress

# 

# Process Description

Author: Jay Patel, AJ Hancher, Brandon Toups, Justin Greenfield, Mina Narayanan

**Version Control:** We’re using Git for version control. IBM Cloud works extensively with GitLabs for version control on projects.

**Task, Issue Management:** For task and issue management, we have elected to use Trello. This will allow us to view what other people are working on. However, since most of our roles remain relatively separate until later integration, we will not necessarily need to utilize this for now.

**Source Code Management:** We are using the Eclipse IDE for development. Each feature is being maintained as a new branch and merged to the master branch after code review by the team. The frontend and backend are kept in separate repositories hosted in Gitlab through IBM’s cloud services. Since the two pieces communicate through RESTful API requests, they can be developed separately.

**Team Communication:** GroupMe is the primary team communication used within our group. For contacting sponsor, email is primarily used, as well as remote team meetings he hosts with Webex.We also use a discord for quick-access meetings and discussions.

# Design Documentation

Author: Jay Patel, AJ Hancher, Brandon Toups, Justin Greenfield, Mina Narayanan

## Languages

Author: Justin Greenfield

Python and Java are the primary languages that we’re using on this project. We are using Java, Java Server Pages (JSP), Java Servlets, and Node.js to program the back-end services, UI, workflow-engine, and reports. We’ll use Python along with the BeautifulSoup library to automatically retrieve information from a website that holds the data that we want to process.

## Tools

Author: Brandon Toups

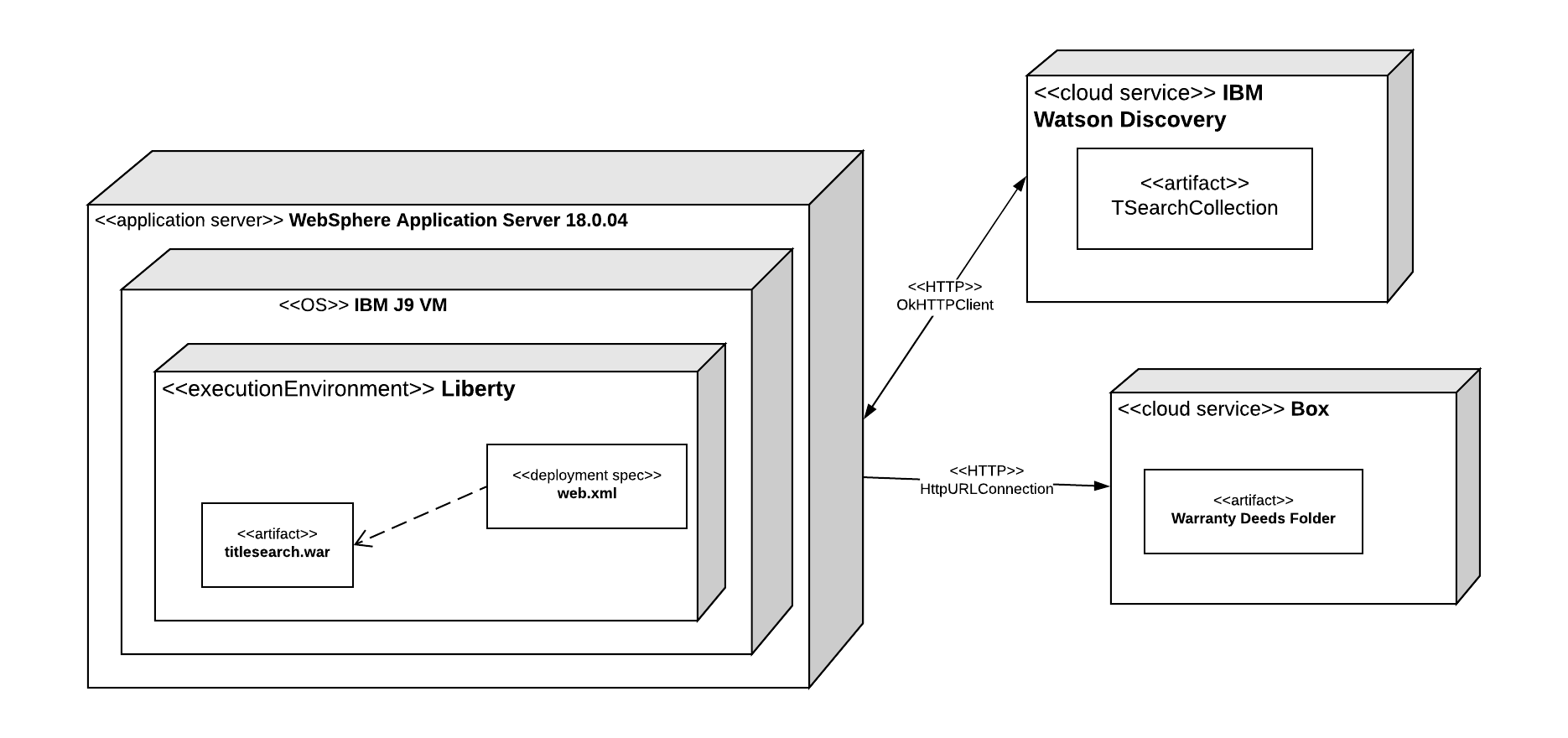
We’ll use IBM Watson Natural Language Understanding (NLU) and Watson Knowledge Studio (WKS) tools to create an AI language model that understands the legal terminology of the documents that we will be processing. Within WKS, we will be developing a corpus, or ground truth, by utilizing the AI to define Grantors, Grantees, and other important entities. To do this, we will be developing our own type system consisting of those entities. To scan the documents provided by the website that provides the warranty deeds, we are using IBM’s Watson Discovery to scan the documents and convert the PDF documents to text using optical character recognition (OCR). We pass the text generated to IBM NLU. Internally, NLU uses the AI language model generated by WKS to understand the Warranty Deed documents.

## UI

Author: Jay Patel

The UI layout consists of a home page with the logo and two input boxes that let the user request a specific warranty deed book and a page number. Both fields are required and validated on the client side as well as the server side. Once the user requests the information, a loading screen appears while a report is being generated with the information. The UI of the report generated is still being designed with the client.

## Architecture



Author: Jay Patel

The web application (.war file) will be hosted on a WebSphere Application Server provided as a hosting service by IBM. The website features and execution is done by the Liberty Runtime running on a IBM Java 9 Virtual Machine. The website’s backend connects to two services: IBM Watson Discovery and Box. The Discovery service provides a storage solution that stores the metadata extracted from scanned document using optical character recognition. However, to use the text extracted from the documents, it has to be stored somewhere to be used to train Watson in the Knowledge Studio. Therefore, we use a content service provider named Box to store the text in .txt files. We are still working on the details of deploying the database and as a result have not shown it in the deployment diagram above.

## Structure

Author: Jay Patel, AJ Hancher

The frontend is provided using JSPs serviced by the backend. Since the UI depends on input from the user, it sends the information provided by the user as a POST request to a Java Servlet. The backend uses these Java Servlets to handle the logic and connect multiple APIs to work together seamlessly. We use the IBM Watson Developer Cloud Java SDK to connect to the IBM Watson Discovery service as well as the other Watson services like Watson Natural Language Understanding and Watson Knowledge Studio. The IBM Java SDK allows you to seamlessly connect to an IBM service since each service provides you with an API key and API URL to target all API requests. To connect to the Box API, we are using the Box Java SDK that provides an easy method to connect to the Box API using JWT Authentication. Each Java Servlet page handles a specific GET/POST request and uses the DiscoveryAPI and BoxAPI java files to establish a communication to the respective APIs and complete the requests. Once the GET/POST requests are completed, the servlet sends the necessary information as a HTTP request to the JSP so it can show any updates in the UI.

Author: Jay Patel, Brandon Toups

The user first submits a query of first name and last name of the property owner inside of a html form on the index.jsp webpage. The form forwards the user submission as a POST request to the PerformTitleSearch servlet that handles the creation of the entire house history. The servlet passes the first name and last name parameters to the TitleSearcherDotComManager Singleton to find the relevant warranty deeds on TitleSearcher.com and return them as a list of WarrantyDeed objects. If the list contains more than one WarrantyDeed, the servlet forwards this list to the warranty/select.jsp webpage to let the user select the correct warranty deed. The WarrantyDeed object is then passed to the WarrantyDeedManager to be analyzed using Watson’s Natural Language Understanding. The WarrantyDeedManager Singleton returns a populated WarrantyDeed object. Finally, the user is presented with a warranty/validate.jsp webpage that shows the pdf and ocr text side by side for comparison along with an input box suggesting the parent warranty deed to obtain next. The ocr text is editable by the user to correct any mistakes.

## 

## 

## 

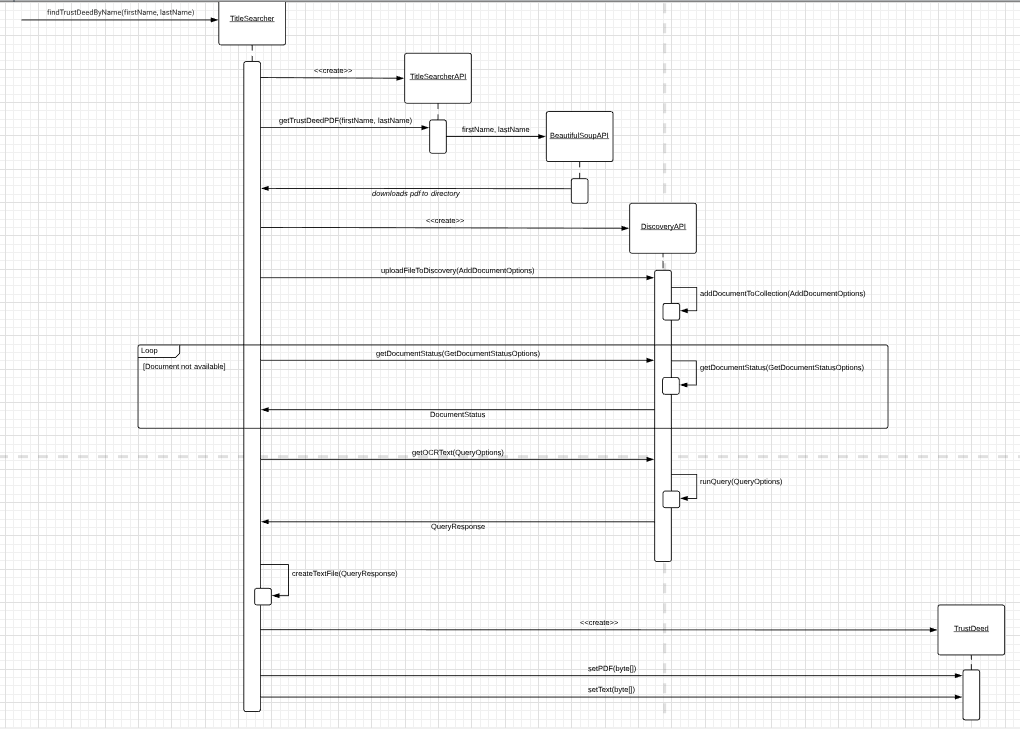
## Sequence Diagrams TitleSearchManager Sequence Diagrams

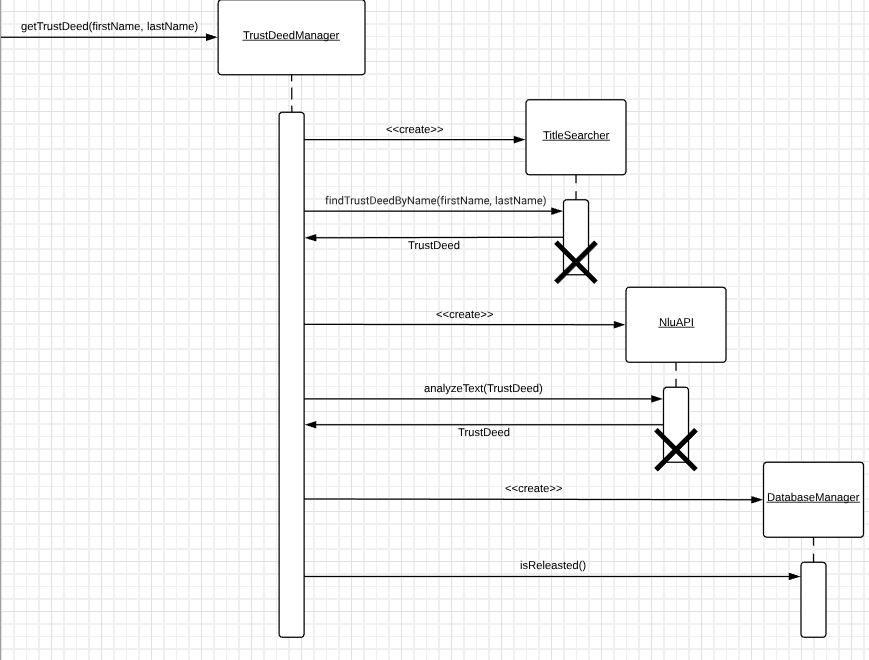
Author: Jay Patel, Justin Greenfield

## 

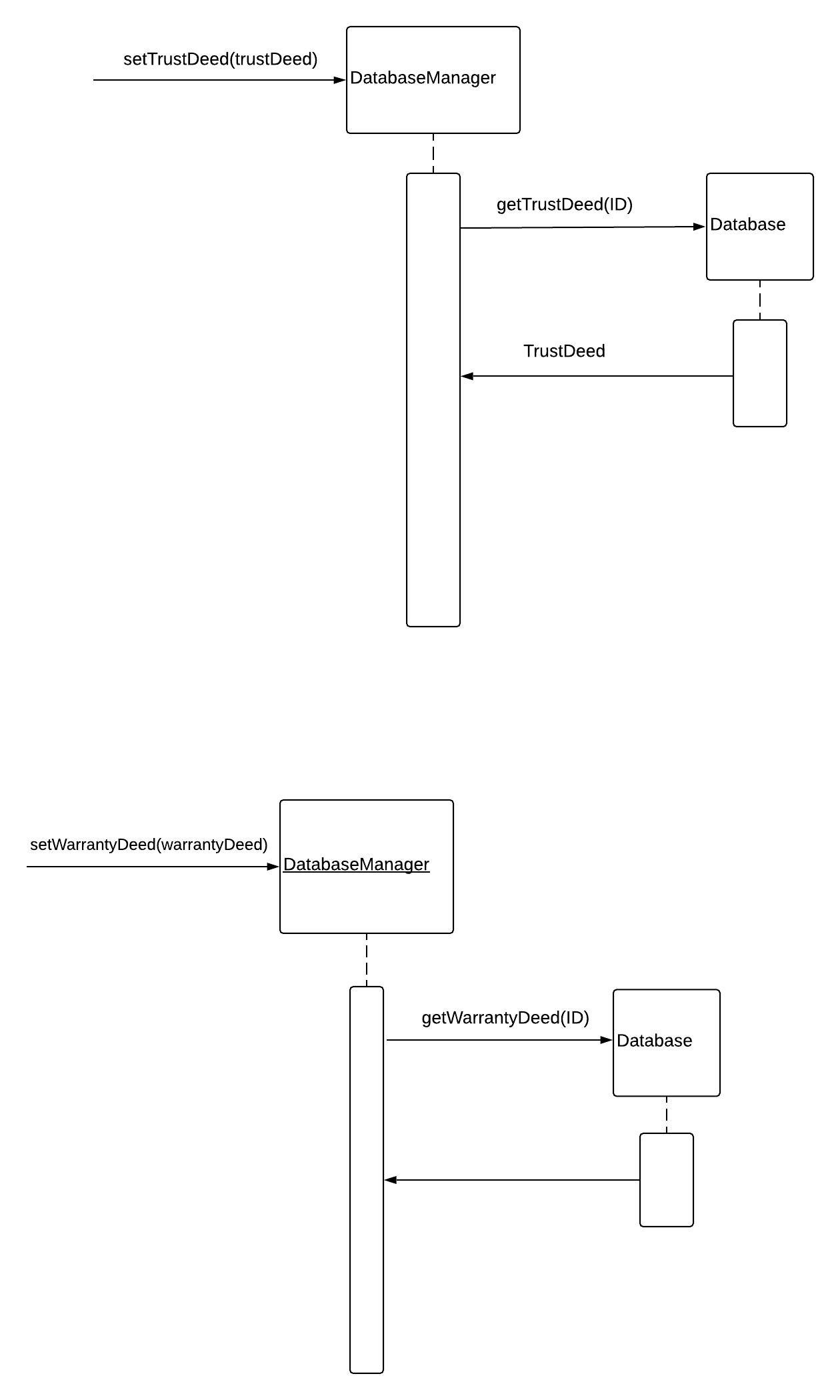
### WarrantyDeedManager and TitleSearcherComManager Sequence Diagrams.

Authors: Jay Patel, Justin Greenfield, Brandon Toups





Database Manager Sequence Diagram (Author: AJ Hancher)



# Management Plan

## Task Assignment

1) Brandon Toups: Understanding Watson Knowledge Studio. How does the model become trained? How does the model extract entities? Creating annotation tasks, working to build folders of bulk pdf files, converting those to txt files using Jay’s Discovery UI, and then downloading those as txt files. Upload these to WKS, then annotate them with entities (relationships come later, coreference chains might not be needed). Development of TrustDeed class, working in conjunction with Justin to build manager for that.

2) Jay Patel: Develop the UI for validating a warranty deed processed using Watson’s NLU. Create the TitleSearcherManager class to orchestrate the recursive searching of the warranty deeds to develop the house history.

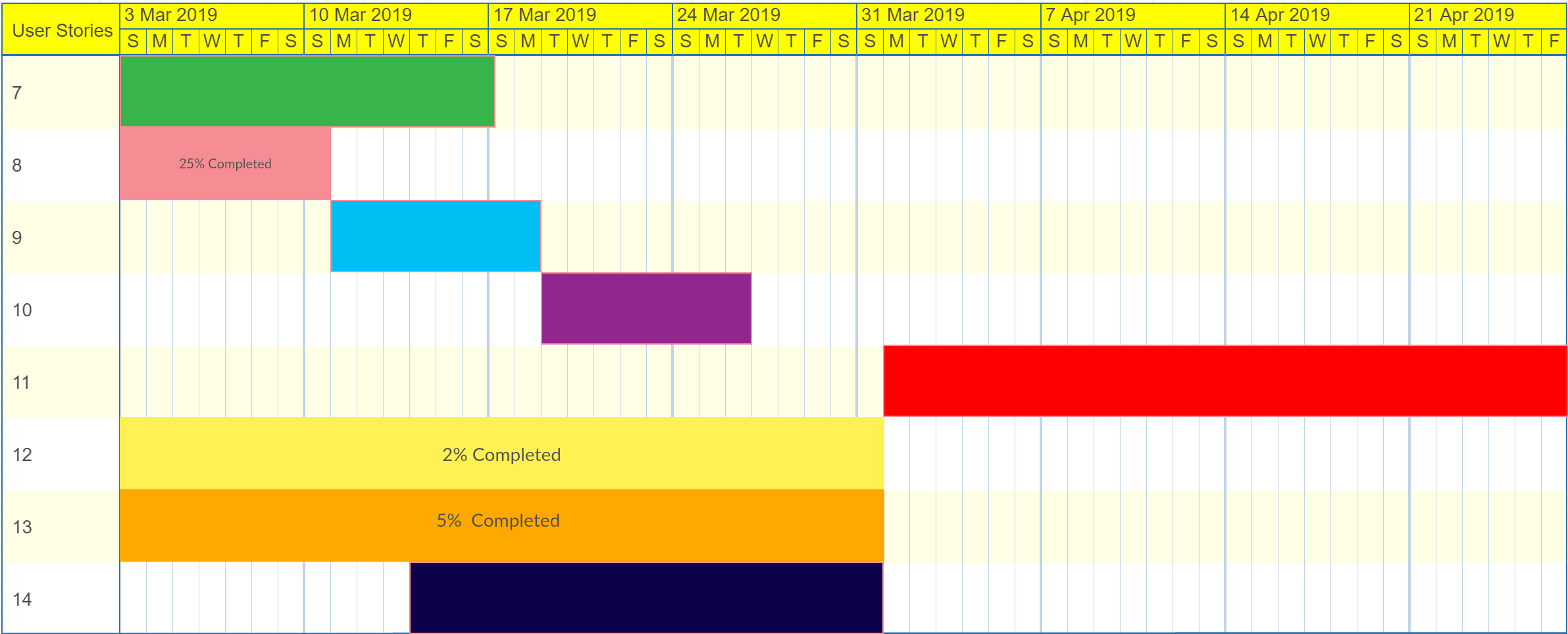
3) AJ Hancher: Development of the Database using IBM Cloudant for MongoDB and the Database Manager Class which contains the get and set functions for warranty and trust deeds. These functions will be called by the other parts of our projects in order to retrieve the correct information for the title search.

4) Justin Greenfield: Development of Natural Language Understanding API, *WarrantyDeed,* and *WarrantyDeedManager* Classes that are responsible for analyzing the document text, extracting the grantors/grantees, and storing the information within the Warranty Deed object.

5) Mina Narayanan: Expansion of BeautifulSoup API functionality to include searching by first and last name and extraction of details from documents that reside on TitleSearcher.com

## Development Schedule

Author: Jay, Brandon, AJ, Mina, Justin



## Planned Code / Feature Freeze

March 1, 2019 is planned code freeze for Cycle 1

# Risk Mitigation

Author: Mina Narayanan

At the encouragement of our sponsor, we restructured our architecture to be a decoupled design.

# Test Plan

Author: Mina Narayanan

We plan to create unit tests for our individual system components during Cycle 2. Afterwards, we plan to implement integration tests to ensure that classes can exchange relevant information, i.e. WarrantyDeed and TrustDeed objects.

# Test Procedure

Authors: Mina Narayanan, Brandon Toups

We have not begun comprehensively testing our system because we have been refining our system architecture and developing an end-to-end prototype for our sponsor. Although we have not established a formal test suite, we have informally exercised our code with the most common user inputs. Most testing is done anecdotally, making sure the pieces build are able to work with the other pieces built.

# Lessons Learned

Author: Mina Narayanan, Brandon Toups

As we’ve progressed our group has learned to plan our meeting times and discussion content ahead of time in order to use our sponsor’s time more efficiently. We’ve organized all of our deliverables in one space and marked important dates (meetings, due dates, etc) on a calendar so we’re all aware of deadlines. We’ve learned about the importance of diagramming software from a high-level perspective to get a better grasp of how to build the system in the most efficient fashion.

Along with this, we have learned that building things separately will inherently yield integration challenges, as different people may understand the root problem differently. To combat this, it is important that everybody openly communicates their understanding of the issues before developing something, as working on a solution for the wrong problem is time consuming and needless.

# Test Documentation

Author: Mina Narayanan

During Cycle 1, testing was limited because we were refining our system architecture. However, we have informally tested our code by trial and error.

# Memoranda

Author: Brandon Toups

## Meeting Minutes

**01/10/2019**

* Email Sponsor
* Drafted email to sponsor, asking for requirements regarding:
  + NDA
  + Non-disclosure
* Review Architectural Spike and other documents
* Review schedule
* Come up with very rudimentary idea of how we want to approach everything
* Cursory review of Watson capabilities

Total Hours: 0.5

**01/14/2019**

* WebEx meeting with sponsor to assign roles, discuss overall project, and introduce us to technology associated with this project.
* Roles assigned:
  + Brandon: Watson Knowledge Studio
  + Mina: Python API and BeautifulSoup library
  + Justin: Watson Natural Language Understanding
  + AJ: back-end functionality and database creation
  + Jay: UI and back-end functionality
* Discuss NDA, non-compete

Total Hours: 1.5

**01/21/2019**

* Gave individual progress reports
* Gameplan for user stories
* Started work on the Architectural Spike report
* Got everybody working on Trello, task management software

Total Hours: 1.5

**01/25/2019**

* WebEx meeting with sponsor to discuss more in depth uses for the programs
* Discuss annotating using Watson Knowledge Studio
* Showed front end to sponsor
* Discuss how all pdfs are different, so no standardization.
* Stopping OCR, now using Watson Discovery

Total Hours: 1

**01/31/2019**

* Group meeting in Senior Design Lab to finish Architectural Spike and share resources
* Discussed later meetings and where our project stands

Total Hours: 1

**02/07/2019**

* Meeting with sponsor
* Further discussions on identifying Grantors and Grantees. Large learning curve for the

Total Hours: 1

**02/19/2019**

* Meeting for first time in-person with sponsor.
* Lots of whiteboard drawings and discussing how each person’s individual parts fit together
* Discuss what integration looks like for our system as a whole

Total hours: 3

**02/24/2019**

* Meeting in library
* Creating architectural diagrams from high-level overview requirements change given by sponsor the week before.
* Must now focus on OOP more, as opposed to merely passing.
* Entire system will be passing Java informational Objects instead of passing bits of information.

Total hours: 3

**02/25/2019**

* Meeting with sponsor
* Went over architectural diagrams and discuss what we did wrong.
* Overall successful. Lot of helpful feedback, and discussions about why the majority of what we have is good. Slight variations necessary.

Total hours: 2

**02/26/2019**

* Classes built *WarrantyDeed.java/TrustDeedjava*, and planning [WarrantyDeed/TrustDeed]Manager.java
* Discuss how the UI will be built and begin working

Total hours: 2

**02/27/2019**

* Group meeting to discuss finishing architectural diagrams for the other parts of the system
* Success overall

Total hours: 1

**02/28/2019**

* Meeting in senior design room to integrate our system parts into monorepo, as opposed to various branches and repos.
* Cycle 1 Report
* Management Plan

Total hours:

## Email Correspondence

----------------------------------------------------------------------------------------------------

Trey Tinnell <ttinnell@us.ibm.com>

Thu 1/17/2019 11:39 AM

AJ Hancher; Brandon Toups; Justin Greenfield; Mina Narayanan; Jay Patel; dvogele@gmail.com

Hey Guys,

Here is a high-level view of what we will be doing in the first phase of our project:

In the first phase of the project, we are going to train Watson to recognize the person selling their property (Grantor) and the person buying the property (Grantee) in legal ownership documents (Warranty Deeds):

**Mina**: Mina will be the one providing sample Warranty Deed document images (Tiff format) to the team (we need hundreds of them). In parallel, Mina will also be working on automating retrieval of documents and other information from the TitleSearcher.com website. This requires Mina to build an API that we can call to perform searches. Searching by name is a given. The real value is having an API where I can pass in a Warranty Deed book and page, or a Trust Deed book and page, and it returns me the image of the legal document and any other related information (which I will describe later).

For our first phase, Mina, we need a search API that lets us pass in a Warranty Deed Book and page number, and it will return the Warranty Deed (basically download it and copy it to a pre-determined location).

**Brandon**: Brandon will use Watson Knowledge Studio to build an AI-based language model that can recognize people selling property (Grantors) and people buying property (Grantees) in Warranty Deeds (legal documents that show ownership). **Note**: This requires the whole team to look at hundreds of Warranty Deed samples (that Mina will provide) and extract paragraphs showing samples of all the different ways that the Grantor and Grantee can be specified in Warranty Deeds.

**Justin**: Brandon's AI-based language model can't work stand-alone. It has to be "plugged in" to Watson Natural Language Understanding (NLU). NLU reads in documents and spits out information about all the entities/nouns, actions/verbs, who receives the action, etc in the document. When we use Brandon's AI-based model with NLU, suddenly NLU can explicitly identify Grantors and Grantees in our Warranty Deeds. NLU spits out JSON about the entire. Justin is responsible for filtering this output and giving us exactly what we need. In this first phase, the only thing we care about is the names of the Grantors and Grantees in the Warranty Deed.

Justin's job is to completely abstract the details of NLU from us. So, for this first phase, Justin should create an API, using Java (coordinate with AJ), that takes in the text of a Warranty Deed document and returns two values....the name of the Grantor and Grantee. If it is unable to determine either of the names, it will return "DONT\_KNOW" as the value for each name it wasn't able to determine. This will help us recognize where we need to improve Watson or the OCR process.

**Jay**: All of the legal documents that represent ownership (Warranty Deeds) or liens (Trust Deeds) are in a TIFF image format (that is, a photograph). All of the valuable information we need to extract to perform our work are in these legal documents. Jay will be converting the photographs of these legal documents into Word or Text files that we can pass into Justin's NLU instance for analysis (NLU can't process photographs) and to retrieve all of the relevant information we need. Jay will need to provide an API where we pass in an image (or a link to where the image resides) and his API will return the Word or Text version of the document (or will write this Word or Text version to a location we specify).

Jay, I will also be sending you a Word document that captures how our reports will look when they are generated. You can take a rough stab at the UI for searches and I will help you refine it.

**AJ**: AJ will be using Java to write the back-end that will be the orchestration engine for making all of the pieces come together.....and the database that stores the document information (e.g., Grantor, Grantee, Warranty Deed Book#, Warranty Deed Page#, etc). AJ, you and Jay will have to coordinate a lot.

**Here's how everyone's work ties together in the first phase:**

1.) Mina provides an API where one can search for a specific Warranty Deed Book # and page # and return a TIFF version of the document.

2.) Jay builds the search UI. One can enter the Warranty Deed Book # and page # in two separate fields.

3.) For WD Book# I enter 123 and for Page# I enter 456

4.) The request goes back to AJ's back-end server.

5.) AJ calls Mina's API, and she returns the associated TIFF image (details TBD).

6.) AJ then passes the TIFF image (details TBD) to Jay's OCR API to convert the image to a Word or Text document.

7.) Once Jay receives the OCR'd Word or Text document, he passes it to Justin's NLU Grantor/Grantee Search API and gets the names of the Grantor and Grantee back. Jay's API calls his NLU instance which leverages Brandon's trained AI-based Watson Knowledge Studio language model under the covers.

8.) AJ then calls Jay's Java Server Page (JSP) page to display the result.

In phase two, we will expand this to extract the previous Warranty Deed Book # and Page # from the legal description of the property that resides in the Warranty Deed. This is where the Grantor, who is selling the property, originally purchased the property from someone else.

Exciting stuff!

Lastly, I want to introduce everyone to David Vogele, who I have copied on this email. David will be here to help answer any questions and we will all be building this together. David is also an UI usability expert, so he will be tremendous help to you, Jay.

Best Regards,

Trey

**IBM**

**Trey Tinnell**

**North America AI Technical Leader**

**IBM Cloud**

**612-308-5634 (Cell)**

**ttinnell@us.ibm.com**

----------------------------------------------------------------------------------------------------

Hey Mina,

Thanks for the response!

A couple of things:

1.) I've been having you use BeautifulSoup because for everything from 1993 - Present, TitleSearcher.com has already extracted out

the Grantor, Grantee, links to Warranty and Trust deeds, and other information.

a.) So, for everything 1993-Present, the idea is that you would go ahead

and fill in the Grantor and Grantee information in the WarrantyDeed object

you return, and somehow flag that this information has already been extracted so

that Watson doesn't have to extract it.

b.) Humphreys County has 1993-present digitized, but other counties will have different

date ranges, so the flag should be something kind of generic like: isDataPreExtracted, which

can have a value of True or False.

2.) Since the goal is to expand to other States and Counties, the state and county needs to be passed in as a parameter

to all of your functions. I suggest you declare the States and Counties as constants. For example:

int ALABAMA = 0

int TENNESSEE = 1

int HUMPHREYS = 0

int DICKSON = 1

Then, calls to your service would be something like:

WarrantDeed myWarrantyDeed = findWarrantyDeedByBookAndTitle(TENNESSEE, HUMPHREYS, 120, 3)

3.) Pre-1993 Warranty and Trust Deeds are the Team's biggest challenge. That is because prior to 1993,

TitleSearcher hasn't done all of the hard work for us. The pre-1993 Warranty and Trust deeds can

span multiple pages, and multiple Warranty or Trust Deeds can be on the same page. For example, a

given Warranty Deed can span 3 pages, but the ending of another Warranty Deed can be at the top of

the first page, and an even different Warranty Deed can be starting at the end of the last page.

We have to figure out how to handle this.

Regards,

Trey

**IBM**

**Trey Tinnell**

**North America Watson, AI, & Deep Learning Leader**

**IBM Healthcare & Public Sector**

**612-308-5634 (Cell)**

**ttinnell@us.ibm.com**

----------------------------------------------------------------------------------------------------

Hey Guys,

It was nice meeting everyone yesterday. It felt like we accomplished a lot during our time together.

I was thinking last night about the project. In the past, I have tried not to be too prescriptive about what my Senior Design teams should be doing, but I'm starting to rethink that strategy. I always felt the hands-on experience of working on a project was the most important thing, but from our meeting yesterday, I really think that the most important aspect of the Senior Design Project is getting mentoring on how to successfully tackle a project. So, with that in mind, I'm going to help show you guys how to successfully deliver a project.

I'm thinking two things. I wish I had more face-to-face time with the team to whiteboard, but that's difficult since I'm only here when I'm guest teaching for Rodrigo's Cloud class, so we will have to improvise:

1.) We need to have more WebEx meetings where I can whiteboard and we can discuss the project as a team.

2.). I probably need frequent 1-on-1 meetings separately with each member so we can discuss the specifics of the pieces you own.

In the mean time, I wanted to follow up on our discussion yesterday about good software engineering and decoupling designs:

1.) We need to take an object-oriented approach to our software.

2.) Everyone's code needs to be integrated, built, and deployed as one unit, from one repository. I can create that in my own private GitHub account.

QUESTION (Please Respond!): DOES EVERYONE KNOW HOW TO DO OBJECT-ORIENTED ANALYSIS & DESIGN??

Back to the object-oriented approach:

1.) We need objects/classes that reflect what we are working on. Thus, we need:

1.) A TitleSearcher class. One calls the methods of this class to perform a search on TitleSearcher.com

- Mina is responsible for this class.

- It should have functions like:

findWarrantyDeedByBookandPage(book#, page#), which returns a WarrantyDeed object.

findWarrantyDeedByName(firstName, lastName), and return a WarrantyDeed.

findTrustDeedByBookandPage(book#, page#), which returns a TrustDeed object.

- CALLING METHODS ON THIS CLASS SHOULD BE THE ONLY WAY TO ACCESS TITLESEARCHER.COM

- Also, this class will abstract away whether the data retrieved came from TitleSearcher.com or our own internal database.

2.) A WarrantyDeed class that holds the data about an individual Warranty Deed.

- Justin & Brandon are responsible for this class.

- holds information such as, Book#, Page#, Grantor, Grantee, ownership chain, etc.

- should have functions such as:

getBookNumber()

setBookNumber(int)

getPageNumber()

setPageNumber(int)

getGrantor()

setGrantor(String)

getGrantee()

setGrantee(String)

getPDF()

setPDF(PDF)

getWarrantyDeedText()

etc.

- WarrantyDeed objects are the only way Warranty Deed information is passed around our system

- Basically, just an information object.

3.) A TrustDeed class that knows everything about an individual Trust Deed.

- Justin and Brandon are responsible for this class.

- holds information such as Book#, Page#, name of owner, etc.

- it will have similar methods to WarrantyDeed class

- should have a isReleased() function that tells us if the Trust Deed has been "released".

- should have a getRelease() function that returns a PDF of the Release.

4.) A WarrantyDeedManager class

- Justin and Brandon are responsible for this class.

- This class should abstract away and hide the use Watson NLU and Watson Knowledge Studio.

- Maybe this class and the TrustDeedManager class should be the only two classes that can access and use the

TitleSearcher class.

- This class should be responsible for storing and retrieving WarrantyDeeds (via the DatabaseManager class),

even though the DatabaseManager functions will only take WarrantyDeed & TrustDeed objects anyway.

5.) A TrustDeedManager class

- Justin and Brandon are responsible for this class.

- Similar to WarrantyDeedManager.

6.) A DatabaseManager class

- AJ is responsible for this class.

- This class should hide everything about the database.

- I don't want anyone to have to need to know or have any dependency on what type of databases is being used under the covers.

- Should have functions for retrieving Warranty Deeds and Trust Deeds from the database.

- Functions should only accept or return Warranty or Trust Deed objects.

- e.g., setWarrantyDeed(WarrantyDeed)

getWarrantyDeed(ID) // This is that ID I've been saying we need to create. Book&Page aren't unique.

setTrustDeed(TrustDeed)

getTrustDeed(ID)

7.) A TitleSearchManager class that is the high-level interface and handles all the orchestration

- Jay is responsible for this class.

- This class should also check to see the DatabaseManager to see if a WarrantyDeed or TrustDeed already exists before calling Mina's class.

- This class should have all the high-level interfaces:

- findWarrantyDeedByBookandPage(book#, page#)

- findWarrantyDeedByName(firstName, lastName) // Can return multiple records. May need to page results.

- findTrustDeedByBookandPage(book#, page#)

- performTitleSearchAbstraction(firstName, lastName)

- performTitleSearchAbstraction(warrantyBook#, warrantyBookPage#)

- etc.

**NOTE: Notice how this class has similar methods to Mina's TitleSearcher class! The difference is that**

**this class checks with the DatabaseManager to see if the Warranty/Trust Deed already exists in our system.**

**If it doesn't, it calls Mina's TitleSearcher class. This completely hides and decouples the fact that we use**

**TitleSearcher.com under the covers.**

8) Jay, all JSPs will only deal with WarrantyDeed or TrustDeed objects. Not with strings or any other parameters...only objects.

**Guys, I need you to work together draw some Sequence Diagrams that show how these classes will coordinate to handle all the different types of requests and scenarios to validate how everything will work and tie in together. Please send these to me. There are still LOTs of details to work out!!!!**

**This object-oriented approach will decouple the design and allow everyone to work in parallel. It will hide the details of the database, Watson, and where the data comes from. It also makes it easier to extend the design when we want to add additional capabilities. This OO design will also ensure we can complete the project on time.**

Regards,

Trey

**IBM**

**Trey Tinnell**

**North America Watson, AI, & Deep Learning Leader**

**IBM Healthcare & Public Sector**

**612-308-5634 (Cell)**

**ttinnell@us.ibm.com**

----------------------------------------------------------------------------------------------------

Hey Brandon,

Good to hear from you! I'm glad you've been interviewing and getting offers. Your future is the exciting thing and the most important thing.

Let's do this in an iterative approach. For this iteration, we will only train Watson to recognize Grantors and Grantees. Here are a couple of thoughts of how to approach this:

1.) I would start with a copy of about 50 warranty deed documents (ideally, use samples with chains of ownership, just 3 owners deep should be fine for now)

2.) I would cut out everything in each document but the paragraph(s) that mention the transfer of the property from the Grantors to the Grantees.

3.) I would ignore relationships for the moment.

4.) I would ignore coreferences for the moment.

5.) We will only train Watson to recognize Grantors and Grantees. Watson is smart. We want to have as many variations of

Grantors and Grantees being specified in a Warranty Deed as possible. Once we show Watson all these variations and

explicitly identify the Grantors and Grantees in these paragraphs, Watson should be able to extrapolate from what it has

learned and start being able to recognize the Grantors and Grantees from any future Warranty Deeds it sees.

6.) After you have trained WKS, then you can associate the machine-learning NLP model with NLU. You should then be

able to pass warranty deed documents to NLU, and it should be pretty good at picking out the Grantors and Grantees.

Test this by passing in the 50 Warranty deed documents to NLU and looking at the result. If the accuracy isn't good enough,

it probably means we need to use more samples for training, identify entities that aren't Grantors and Grantees when we

train WKS, and finally possibly consider relationships and coreferences. We will take the simplest approach first.

Looking ahead, in our next iteration, we will tackle seeing WD Book#'s and Page#'s in the documents. I would tackle this by creating a rules-based annotator that uses regular expressions to pick out Book & Page numbers. One then pre-annotates the training documents with the rules-based annotator before doing the manual annotations.

Regards,

Trey

**IBM**

**Trey Tinnell**

**North America Watson, AI, & Deep Learning Leader**

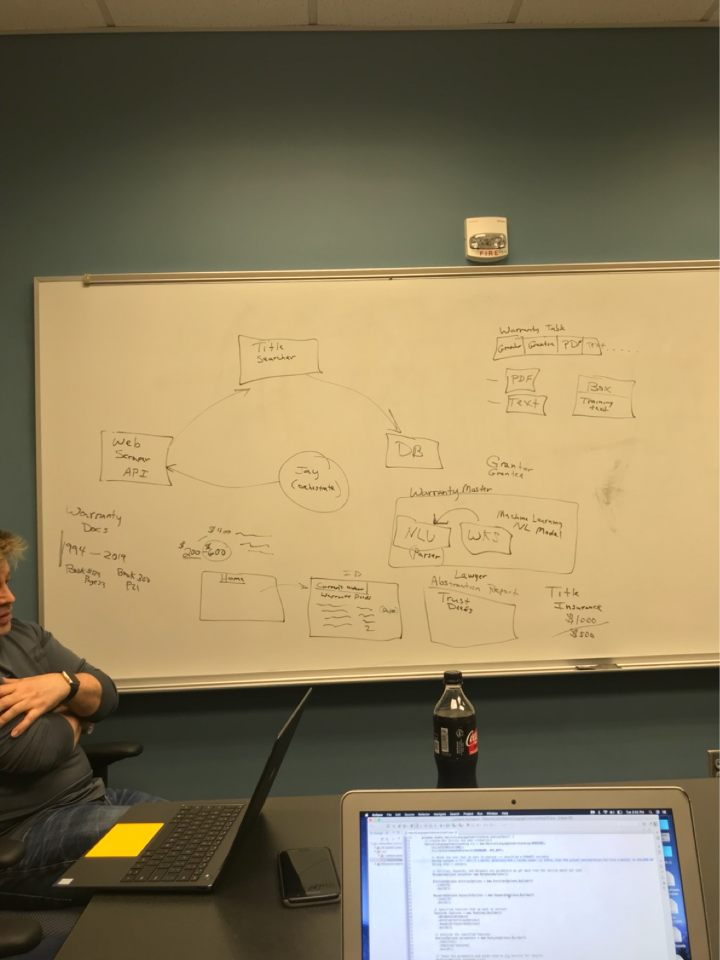
**IBM Healthcare & Public Sector**

**612-308-5634 (Cell)**

**ttinnell@us.ibm.com**

----------------------------------------------------------------------------------------------------

## Whiteboard Pictures



# Version Description

Version 1.1

Description: The application currently can search using book number and page number. The frontend UI consists of a home page where the user can input a book number and page number. The backend is able to process the input by the user to grab a warranty deed from TitleSearcher.com and pass it to IBM Discovery to be read using Optical Character Recognition. The pdf and text obtained through OCR are then stored on separate folders on Box.

Key features: Obtain warranty deed pdf and text by searching through the home page.

Known bugs/issues: The IBM website only allows the python script file to run once and keeping the script running continuously creates confusing errors on the servers.