**CIOBrain Deployment**

Detailed Design Documentation

Anton Horvath

Anuja Sahu

Scott Lorance

Kevin Lieng

Jose Colina Salas

Saige Wright

# Abstract

This detailed design document introduces the GUI, static model, and dynamic model of the CIOBrain project. Note that this document only gives additional details on the existing design of the project. The GUI includes visual aspects added or changed in the project. The static model includes class diagrams that show changes to existing components and new components. The dynamic model comprises sequence diagrams that show the additional interaction or change of interaction with the product. Finally, this document includes a traceability model that maps the design elements back to the requirements.

# Table of Contents

[**Abstract**](#_x6rwfpx8ibno) **2**

[**Table of Contents**](#_oi6ud0wjlf0m) **3**

[**List of Figures**](#_n5yatnf7llxl) **4**

[**List of Tables**](#_f9g83v8zvuly) **4**

[**Introduction**](#_fol7u981ugxu) **5**

[**GUI (Graphical User Interface) Design**](#_j2t9pkqxa6d3) **6**

[MSI Installer](#_r2owj890bq8v) 6

[CIOBrain Desktop Application Login Authentication](#_krhvbzddzudc) 7

[**Static Model**](#_qqfah0gjw8pl) **8**

[MSI Installer](#_lrk4oifoz55o) 8

[CIOBrain Desktop Application Login Authentication](#_n20ghfqs6kza) 9

[**Dynamic Model**](#_d6ix499mx3lp) **11**

[MSI Installer](#_j9k3if4nd4ys) 11

[Azure Authentication](#_taxegj90evcq) 12

[Azure Deployment Script](#_afo8y96ji1ul) 13

[Data Transfer](#_c60dadgfxtwg) 13

[**Rationale for Detailed Design Model**](#_8jsfvbtr8717) **14**

[**Traceability from Requirements to Detailed Design Model**](#_h0i4gexv9wfg) **15**

[**Evidence the Design Model Has Been Placed Under Configuration Management**](#_695fev9d0xlq) **16**

[**References**](#_kb0iqtnj5ixl) **17**

# List of Figures

Figure 16

Figure 2 6

Figure 37

Figure 4 7

Figure 5 8

Figure 6 9

Figure 7 10

Figure 8 11

Figure 9 12

Figure 10 13

Figure 11 13

Figure 12 16

# List of Tables

Table 115

Table 2 15

# Introduction

The purpose of the detailed design document for CIOBrain is to give a detailed overview of the design of the CIOBrain product. Since we are working on adding onto an already made software, this detailed design document will focus on the additional designs and aspects.

The scope of the detailed design document includes the following:

* GUI (Graphical User Interface) Design  
  This section shows the GUI design of the CIOBrain features being implemented this semester.
* Static Model  
  This section shows the static model of the CIObrain product by showing the details of the class diagrams.
* Dynamic Model  
  This section shows the dynamic model of the CIObrain product by showing the details of the sequence diagrams.
* Rationale for Detailed Design Model  
  This section gives the reasoning and rationale behind why the class and sequence diagrams were chosen.
* Traceability from Requirements to Detailed Design Model

# GUI (Graphical User Interface) Design

## MSI Installer

Requirement 1: The desktop app should have a .msi installer file that can be run to instantly install the application to the Windows machine. A desktop shortcut should be created where the user can double-click to launch the application.

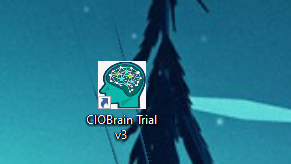


Figure 1: CIOBrain Trial v3 Desktop Shortcut

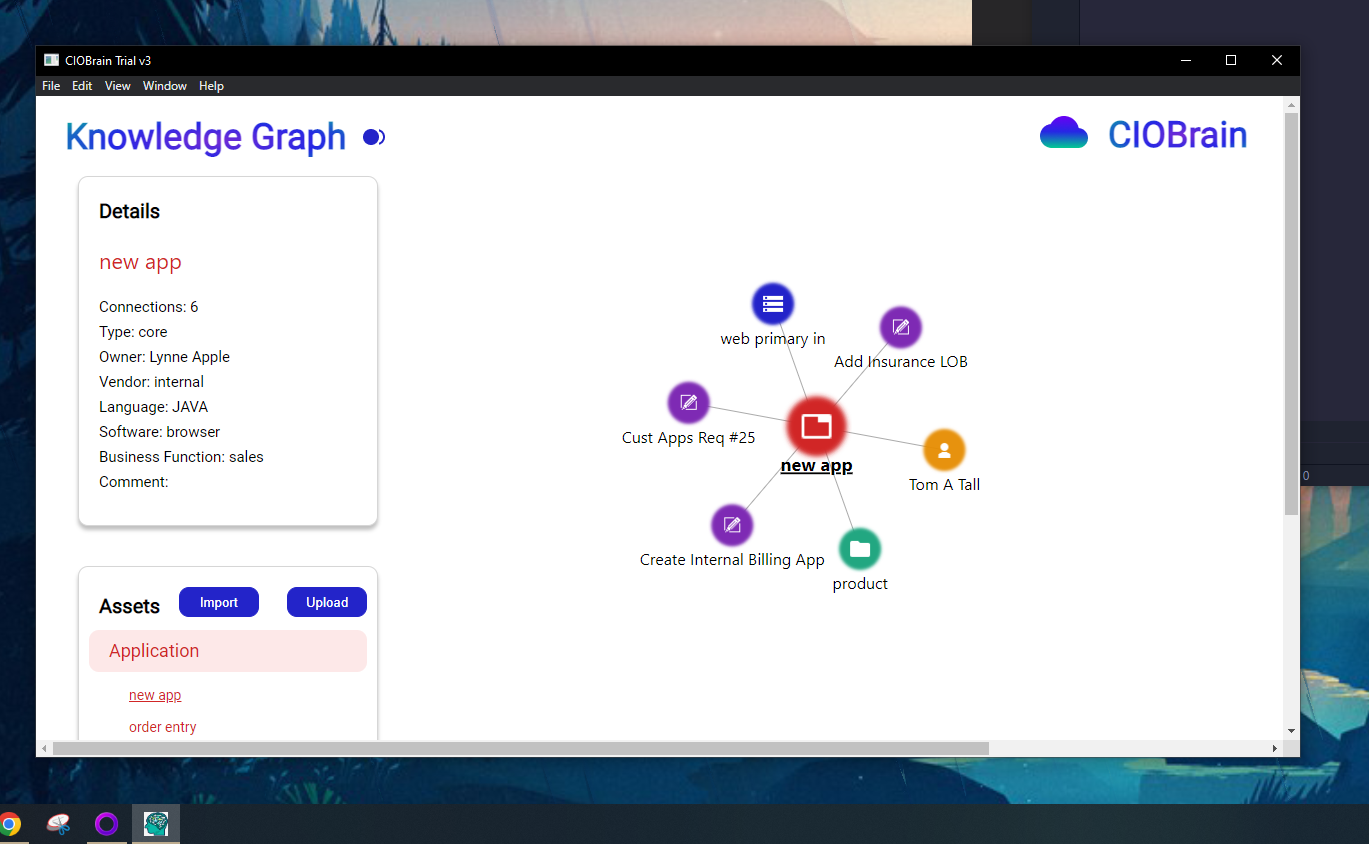


Figure 2: CIOBrain Trial v3 Running on Native Process

## CIOBrain Desktop Application Login Authentication

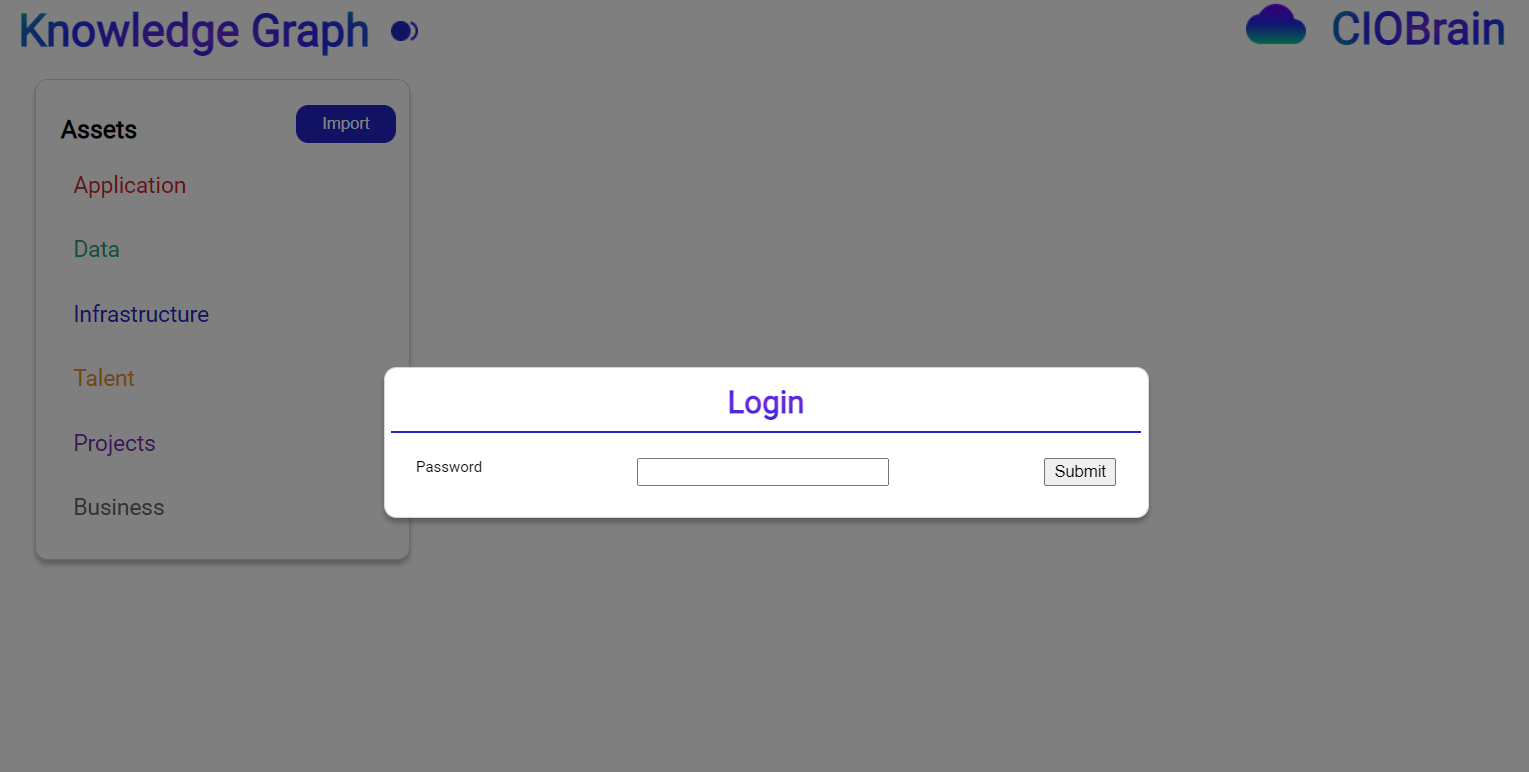


Figure 3: Login Function Visualization

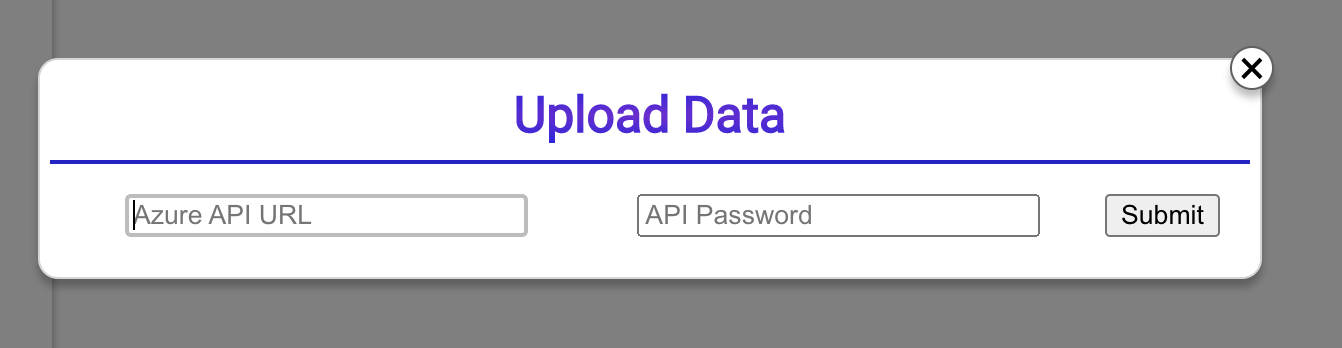


Figure 4: Data transfer capability GUI

# Static Model

## MSI Installer

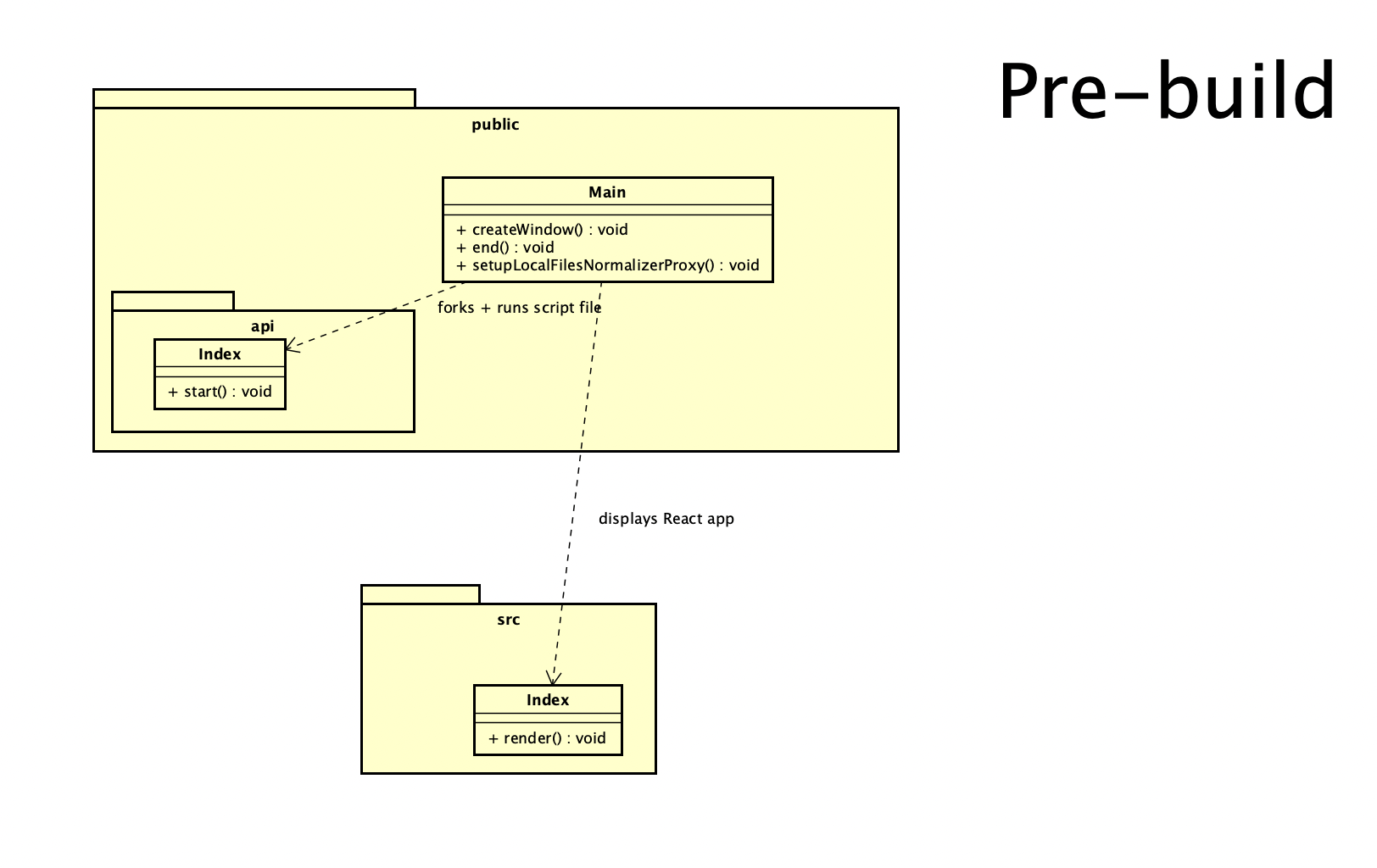


Figure 5: Class diagram pre-build for MSI Installer Wrapper

## CIOBrain Desktop Application Login Authentication

## 

Figure 6: Class diagram pre-build for Desktop Application Login Authentication

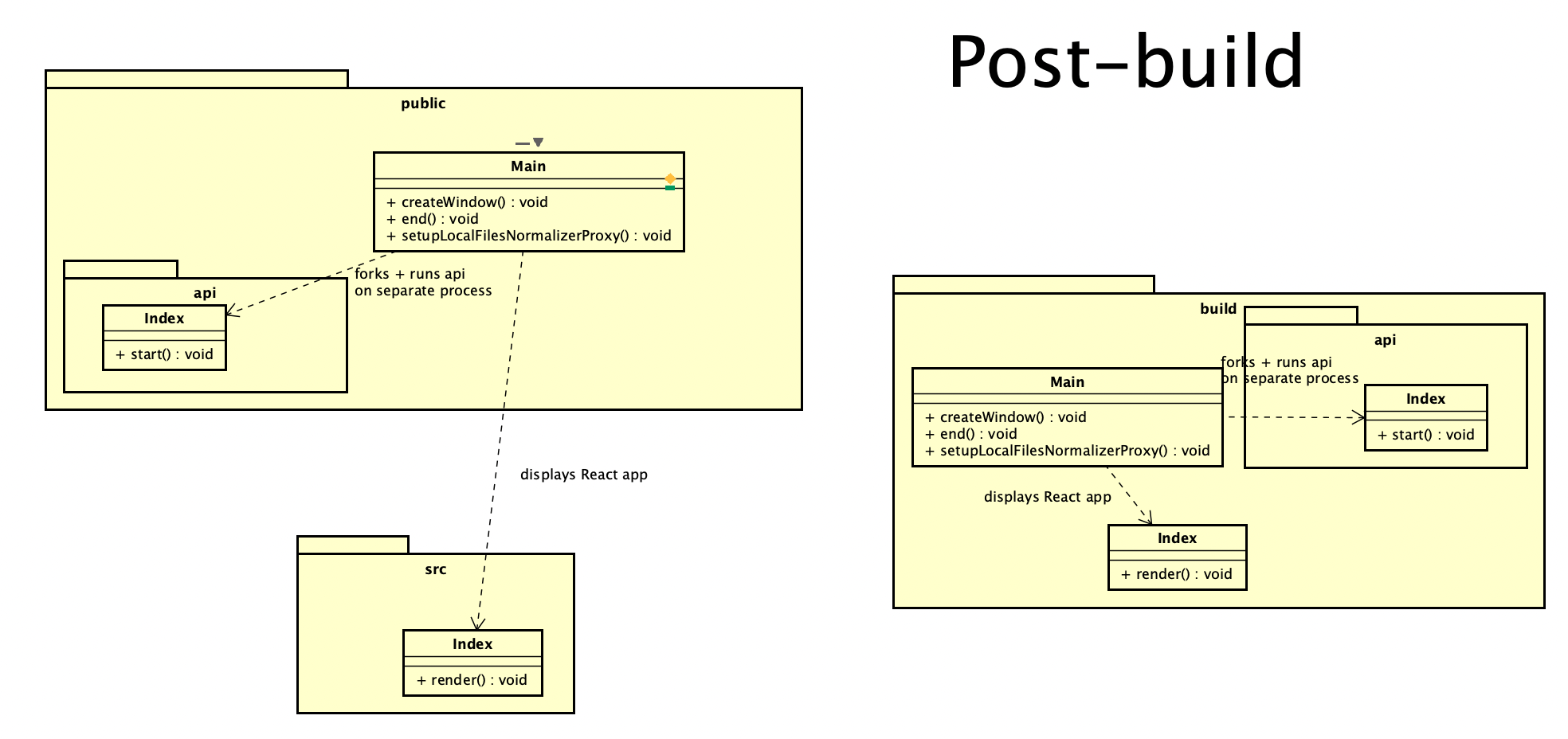


Figure 7: Class diagram post-build for MSI Installer Wrapper

.

# Dynamic Model

## MSI Installer

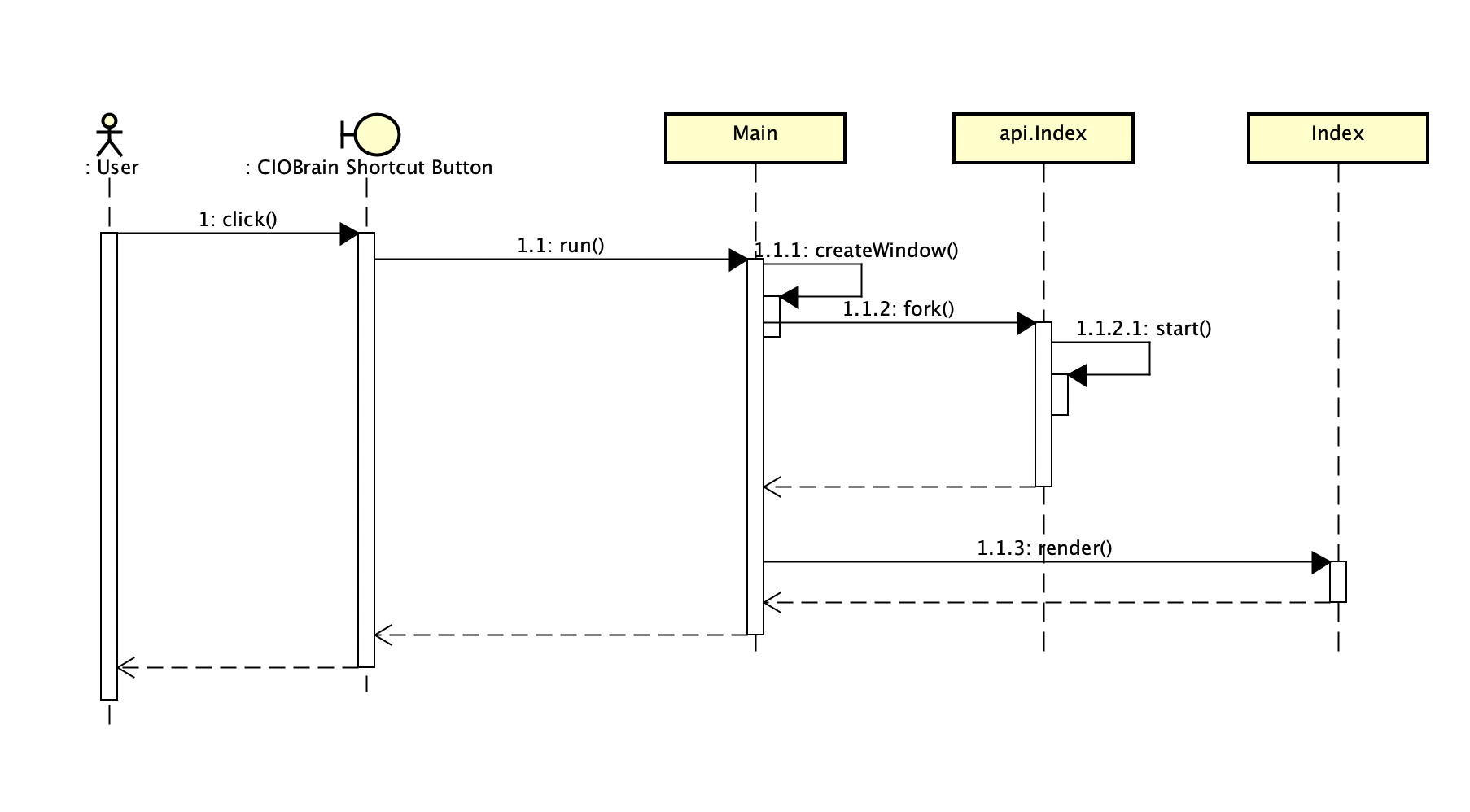


Figure 8: Sequence diagram for MSI-Installed Application Run

## Azure Authentication

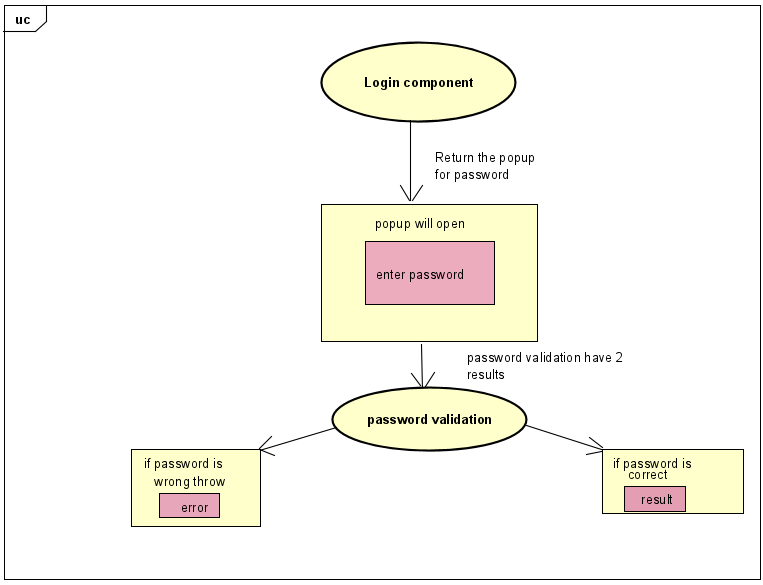


Figure 9: Login Authentication Diagram

## Azure Deployment Script

Figure 10: Azure Deployment Communication Diagram

## Data Transfer

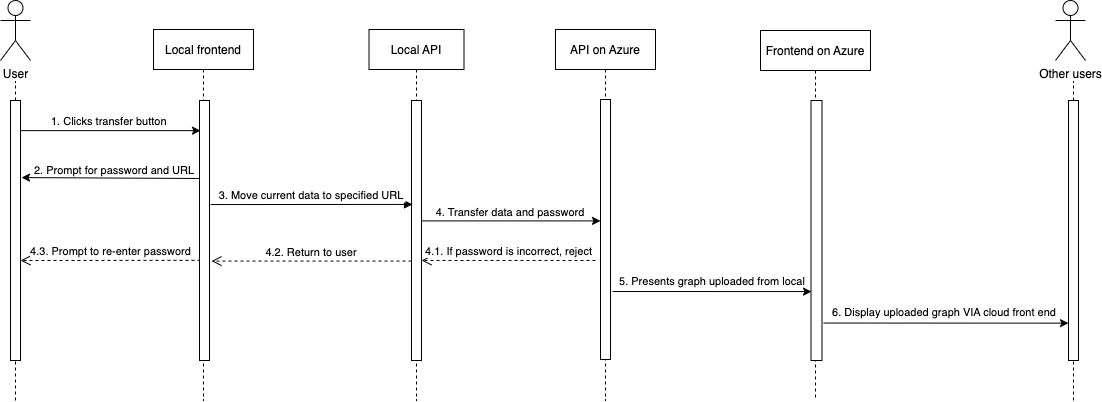


Figure 11: Sequence Diagram for data transfer from the desktop app to Azure

# Rationale for Detailed Design Model

The MSI installer wrapper had to combine both existing web applications for CIOBrain and create a new application that would launch both at the same time to operate offline on a local machine. This could be done using Electron.js, a framework that uses a Main class to launch a web application running locally. Since a client-server architecture was still necessary to keep the app’s functionality intact, the API application instance was packaged into this Electron application and run on a separate process upon startup. The main process where the Electron app was running would display the React application, achieving the client-server architecture packaged into a single app. This design enabled requirement 1’s implementation.

The rationale for the easy azure deployment script is to fulfill requirement requirement 2 for easy transfer to azure, by allowing individuals without azure experience to upload and run the service by inserting a few credentials and running the script. The sequence diagram that shows the process for data transfer from a user’s local machine to the Azure application starts with the user clicking on the transfer button on their local machine, which prompts them to specify the URL at which the data needs to be received at, and the password set by the user. All the currently opened graphs and password are pushed to the Azure server, but if the password entered is incorrect, then the user is prompted to re-enter the password. Otherwise, the server on Azure accepts the files and rejects any duplicates. Once all the data is transferred successfully, Azure API will display the uploaded graph via the cloud front end to any users who use the cloud hosted version.

# Traceability from Requirements to Detailed Design Model

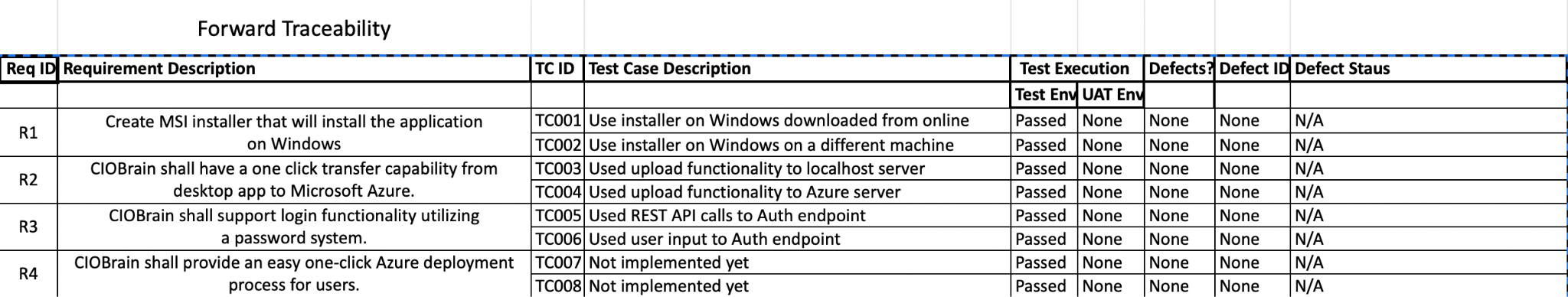


Table 1: Forward traceability

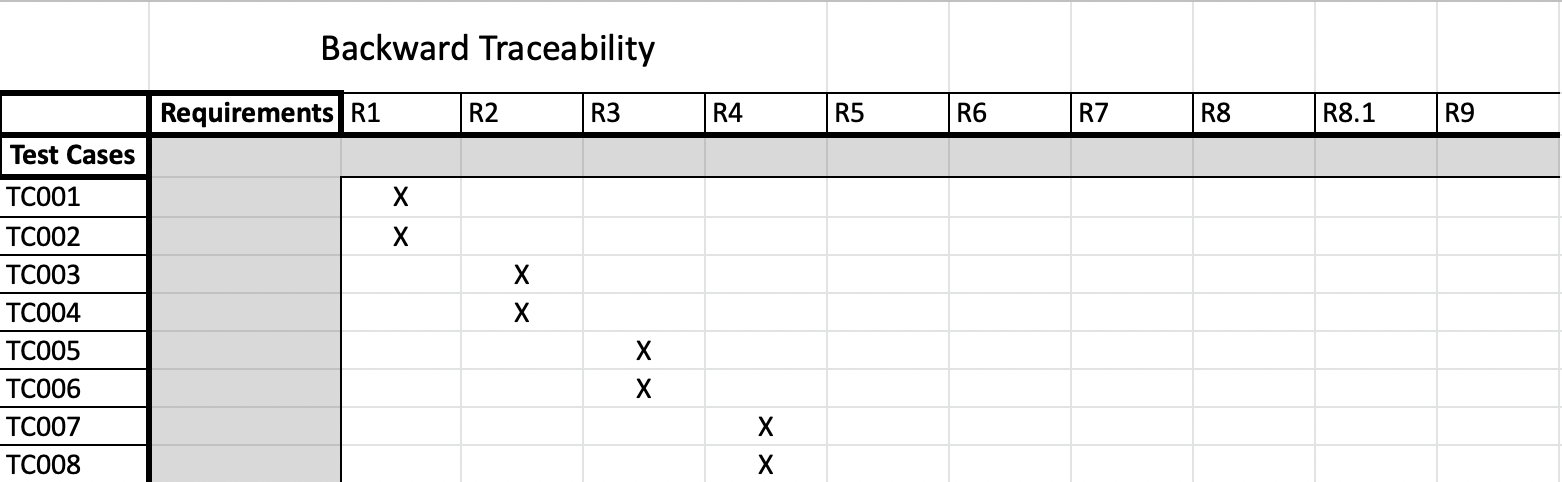


Table 2: Backward traceability

# Evidence the Design Model Has Been Placed Under Configuration Management

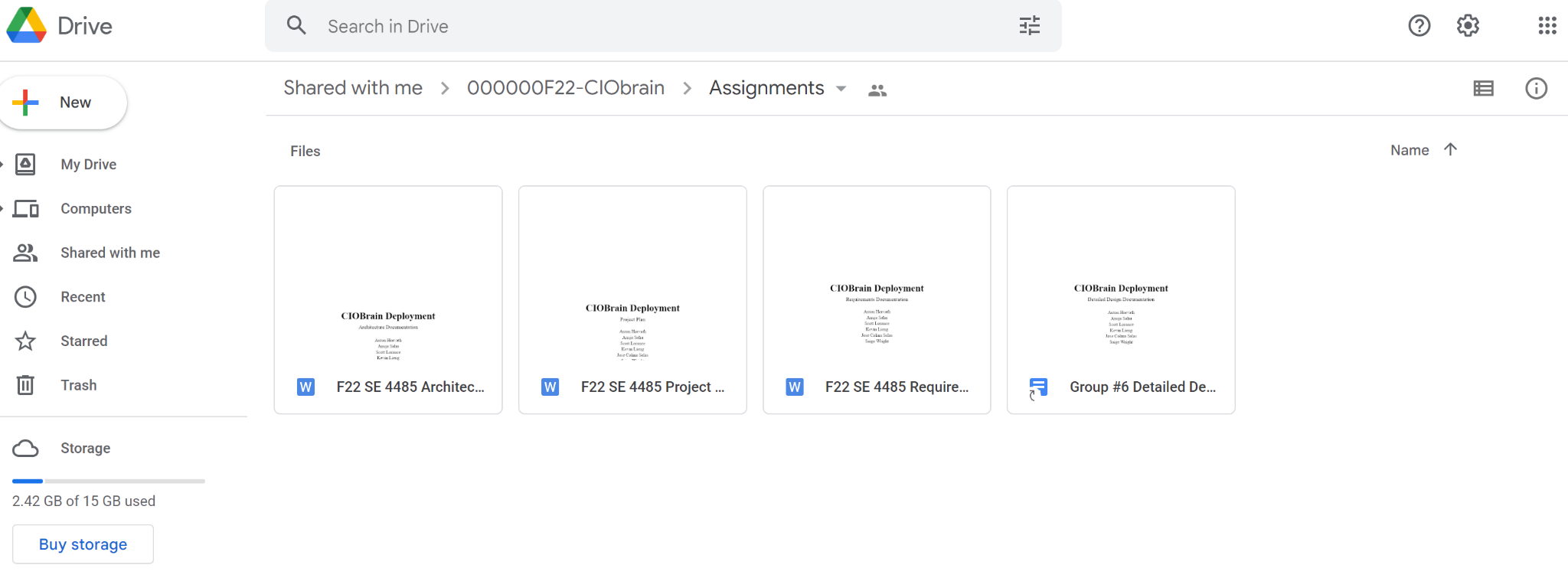


Figure 12: Evidence of Configuration Management

# References

N/A