Software Architecture Specification

March 25, 2020

Team 3- Gracie, Katie, Ben, Abby, Brad, Owen, Ana

1. System and Architectural Context

1.1. **Rationale**

Overall, our architecture consists of an R Shiny application, an Shiny IO server, a SQL database, and a database server (we are using the Data Cats server for our database). We chose to use the R Shiny application with the Shiny IO server because it allows for quick dashboard development and handles a lot of the overhead associated with building an application. We chose to use the Data Cats server for our SQL database because it will be easy to connect with our Shiny IO server. Additionally, the Shiny IO server will allow us to easily deploy our application securely and freely. To visualize our architecture, we chose to use both a module view and a C&C view. Because our application runs on the web, it does not need to run with any particular hardware, so we did not use the software/hardware diagram. The module view and C&C view allows us to best describe the attributes of each class (professor and student) we want to include while demonstrating the database relationships.

1.2. **Scope**

This system covers an instance of the MGS application for one specific course. Because of our limited time for this project, we will not provide a mechanism for creating new users. Instead, users will be pre-set into the database by an admin when an instance of the application is created. For the developers, our chosen architecture will make developing pretty easy, especially with the data cats server. For our users, the Shiny application pre-made UI components will make the application user friends. However, the use of Shiny IO and Data Cats server means that the professor users will need to launch their own instance of the application on their own Shiny IO server to make it available to students.

1.3. **Definitions, Acronyms, and Abbreviations**

1.3.1 ODBC - Open Database Connectivity is a standard applications programming interface (API) for accessing database management systems.

1.3.2 Free TDS: set of libraries for Linux that allows programs to communicate with a Microsoft SQL Server

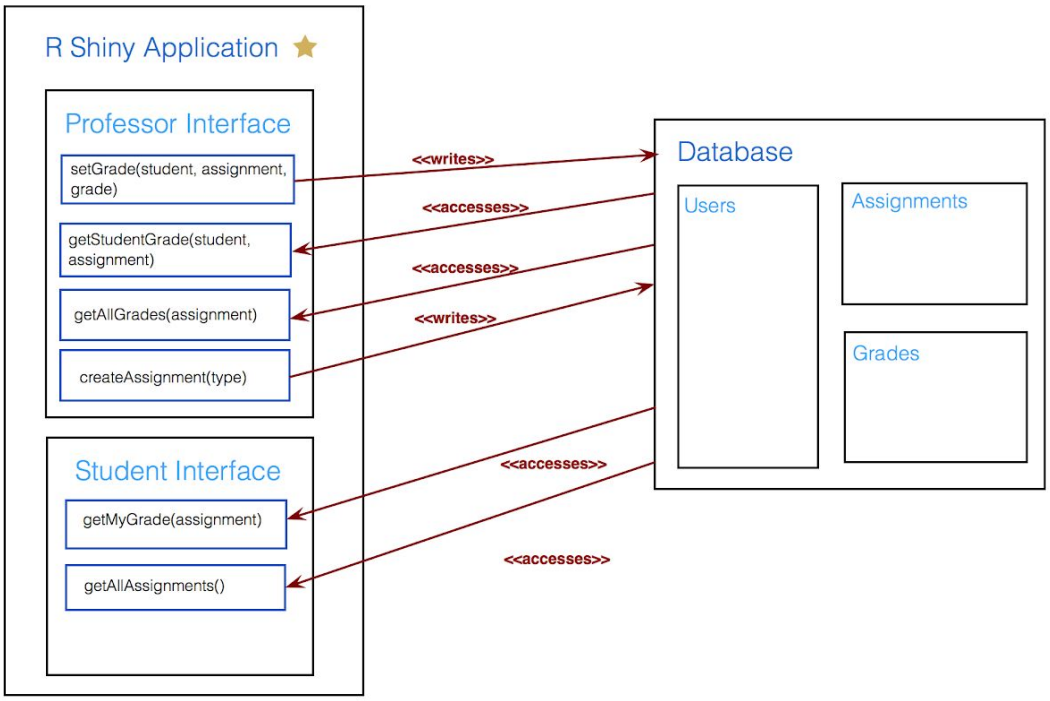
1.4. **Behavior**

At a high level, the R shiny communicates with the Shiny IO server through HTTPS. Then, the Shiny IO server communicates with the database to send information to and from the application. The professor interface has the ability to read and write to the database, while the student interface can only read. The professor and student interfaces communicated only through shared data (ie. the database).

2. Architecture Views

2.1. **Views**

Module View



2.2. **Element Catalog**

Student Interface

a. purpose: read assignment and grade data
b. interface semantics: independent of professor interface, specialization on the R Shiny Application
c. protocols: HTTPS to communicate with database server

d. data form Elements

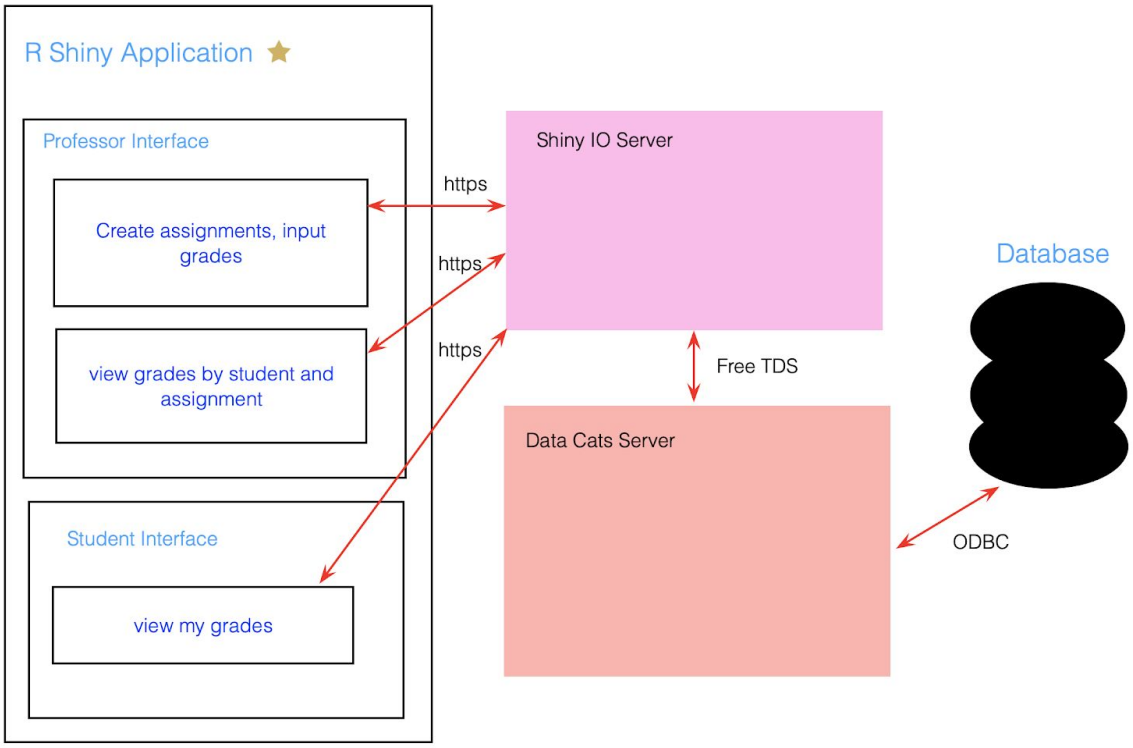
Professor Interface

e. purpose: read and write assignment and grade data
f. interface semantics: independent of student interface, specialization on the R Shiny Application
g. protocols: HTTPS to communicate with database server
h. data formats: receives objects: receives objects from database containing strings and integers

Database

i. the SQL database will handle all concurrency issues
j. data format: SQL tables

Connector & Component View:



Component View:

Professor Interface: view assignments/grades page

i. Purpose: Allows the user to view the assignments and the corresponding grades for all students.

ii. Interface Semantics: connects to Shiny IO server using WiFi or any other internet connection
iii. Protocols: HTTPS - this allows the R Shiny application to securely communicate with the Shiny IO server.
iv. Data formats: receives objects from database containing strings and integers

Student Interface: view grades

v. Purpose: Allows the user to view the grades that have been inputted corresponding to each assignment
vi. Interface Semantics: connects to Shiny IO server over wifi or any other internet connection
vii. Protocols: HTTPS, this allows the R Shiny server to communicate with the Shiny IO server
viii. Data formats: Receives integers and strings from the IO server

Shiny IO Server

ix. Purpose: Hosting of the remote shiny application.
x. Interface Semantics: shiny.io's publishing site
xi. Protocols: Free TDS - this driver allows communication between the Shiny IO server and the Data Cats server.
xii. Data formats: R, HTML, and Javascript code

Data Cats Server

xiii. Purpose: Host our SQL database
xiv. Interface Semantics: SQL within R
xv. Protocols: Free TDS - this driver allows communication between the Shiny IO server and the Data Cats server.
xvi. Data formats: numeric, character, date

Data Base

xvii. Purpose: stores every student's data and all assignment data
xviii. Protocols: ODBC - this connector allows communication between the Data Cats server and the database so the database can be managed.
xix. Data formats: SQL tables

3. Across Views Description

3.1. Views

Module View: This view provides detailed information about the functionality of the professor and student sides of our application while showing the database relationships. This view provides detailed information to developers when to make database memory calls.

Connector & Component View: This view provides information about how all the pieces of our project work together. It gives more specificity to the backend software we are going to use as well as the types of connectors we are going to be using. Data is the only thing being exchanged over each path, represented by the red arrows.

Both these views show many of the same elements in the “R Shiny Application,” though the Module View provides greater detail. Arrows represent other relationships between elements within the MGS system.