Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4316: Senior Design I
Fall 2021

Team GI
ROTC Database

Tochy Egeonu
David Rademacher
Jonathan Banda
Ian McKenzie
Jacob Ryan

Revision History

Revision	Date	Author(s)	Description
1	11.11.2021	Team	Base changes
2	12.7.2021	Ian	Updating for new project
3	4.21.2021	Tochy Egeonu	Revisions

CONTENTS

1 INTRODUCTION

We will build a database backed website to host as a centralized source for the sennior design lab to gather and store information regarding items lent out. The system will need to work in a cyclic mannner allowing a user to identify the student checking in/out, proccess item to be checked in/out, and update the system according. Other features such as report generation and reminders may be implemented to assist the senior design lab in recovering borrowed items from students or take action accordingly.

2 System Overview

The entirety of our system will rely on calls made to the database. Therefore in order to build such system we need the foundation of our architecture to be comprised of a database layer. The database layer plays an important role in maintaining integrity and functionality of the entire system. Working along side the database layer should be a login layer. The login layer serves its purpose in encapsulating/restricting privileges granted to users. This layers is essential in integration of the system. Other than those two layer there is only one other high level layer that must be taken into consideration, which is a dashboard layer. The dashboard layer consists of all the sub modules features of the system. This layer handles the flow and functionality of the systems features. The Layer that will put everything together is the Web Application Layer which in other words is the "api" layer that handles request to and from the database that are made on the web app. It will use the REST framework in order integrate properly.

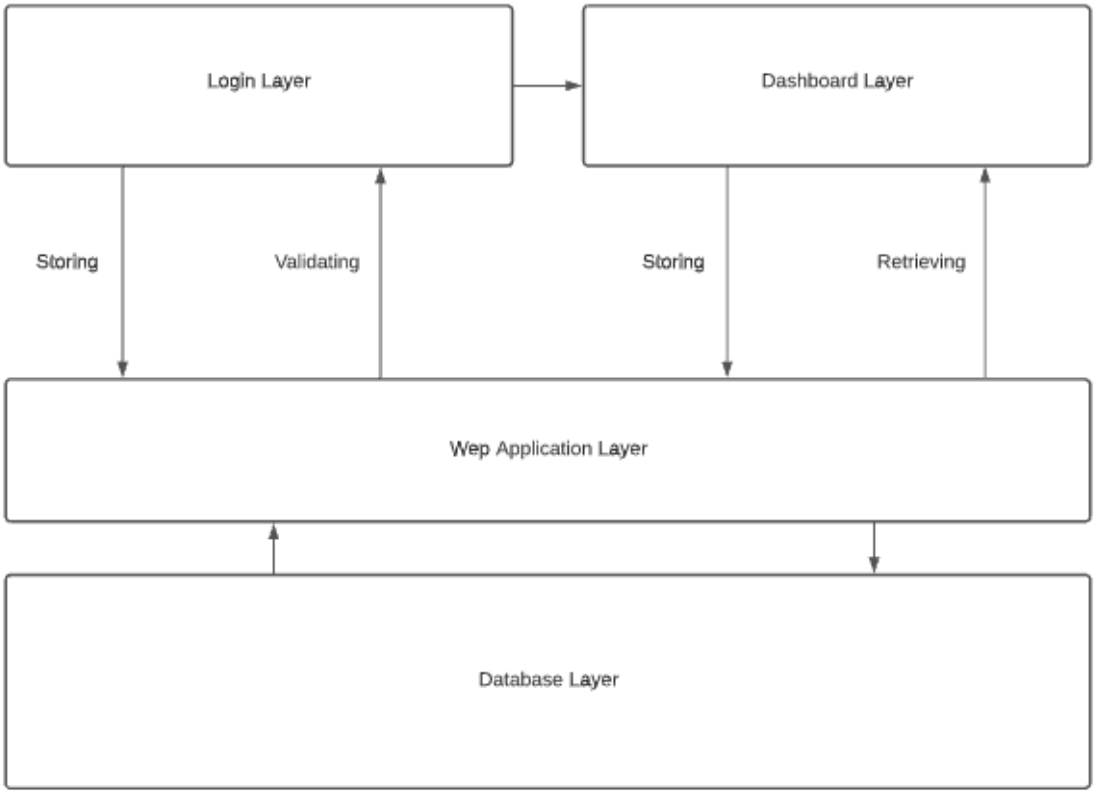

Figure 1: Architectural layer diagram

2.1 LOGIN

The login layers primary function is to handle and control access to the system. This layer is the for-front of the system is responsible for logging users in given their unique email and password. The login layers functionality can be defined as below.

- **Create account**
  - Enter valid email address : username
  - Enter valid password : greater than equal to eight characters with one special character and one capital letter at a minimum
- **Login Users**
  - Validate Username : email
  - Validate Passwords : greater than equal to eight characters with one special character and one capital letter at a minimum
- **Reset password**
  - Enter username (email) for instruction on how to reset password.
- **Dashboard Privilege**
  - Upon login user will have access to the dashboard and other functionality based on rank.

2.2 DASHBOARD

The dashboard layer will handle the flow of functionality. This will be represented as a single view web-page with smaller components. A user will have the ability to navigate between each component. These components will include the following.

- **User analytics**
  - display report of a select individual from the database
    - Name
    - ID
    - Email
    - Inventory checked out
    - Date and time of request
- **Inventory**
  - Check inventory of all items
  - Display inventory
- **Checkout**
  - Find student account in database
  - Add items to cart from database
  - Add items from cart to student account
- **Return**
  - Find student account in database
  - Find items in student account
  - Remove items from student account

2.3 DATABASE

The database will be comprised of a variety of database schemas. Each schema will will be relevant to the component in view. It it important to understand that the database will be hosted by Amazon web services and we will pull/push data as needed. Below is a general outline for the schemas needed in order to provide functionality to the entire system.

- **Credentials**
  - Primary key : user id
  - Password
- **Users**
  - Foreign key : user id
  - Name
  - Email
  - id
  - Grade
- **Inventory**
  - Foreign key : user id
  - Inventory data tuple of items from form

2.4 Web Application Layer

The web application layer is the part of the project that integrates the front-end and back-end together. This layer is what puts everything together, it will be primarily supported through the REST framework that will allow for serialization of data to and from our database. This layer will need to handle the following.

- **Request Handling**
  - This portion of the "api" will need to send data to and from the database. It will need to accept a card swipe and item pair and be able to store such data into the database where it must reside according to user.
- **Page Generation**
  - The views of the entire web application must be seamless in how they operate and according to user privilege. Users that are not allowed to view a page or request data should not have access to such data. Each view can have its own separate url or a combination of a parent url.
- **Database Querying**
  - The data query's will need be serialized as we are going to use react to create the front-end of the project. The serialization converts the python code and data into a much more usable format for the front-end; JSON. And will allow the web page to display or send data between the database and the front-end.
- **User Authentication**
  - The validation of users will be a constant request of permission between each view. Admins should have access all across the system while those with restriction will not. The system must handle making requesting to the database to verify a users unique id and return the access privilege key True or False.

3 Subsystem Definitions & Data Flow

In this section, each layer abstraction is broken down in further detail. The logical subsystems which compose each layer are graphically represented, and their interactions are shown. Each of the **Login**, **Dashboard**, **Web Application**, and **Database** layers has its own modules that represent programming units with specific tasks. A description of the data flow within and between each component is provided in the following sections.

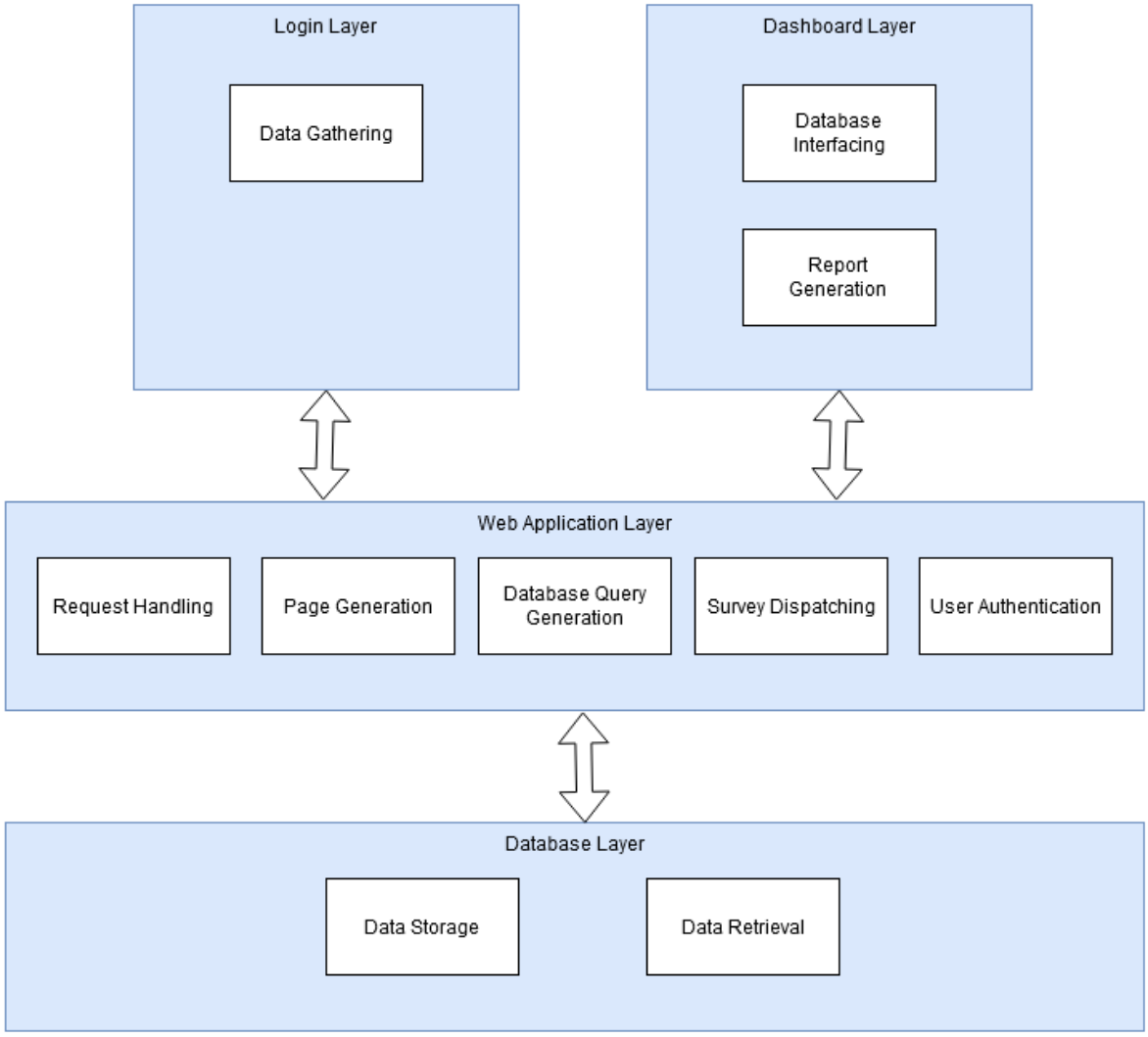

Figure 2: Data flow in the web application

4 Login Layer

This layer is tasked with interfacing with users through forms. Accessibility, layout, interactivity, and security are among the top concerns for the login page.

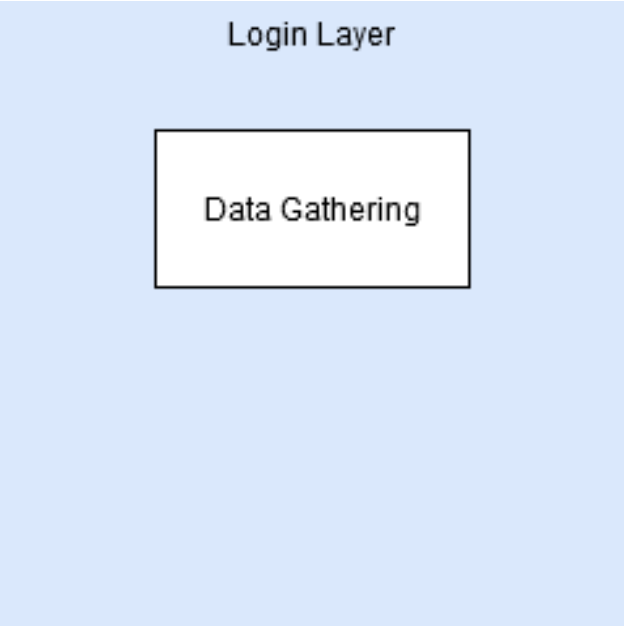

Figure 3: Login subsystem diagram

4.1 Account Acsessability

The primary function of the Login Layer is to gather identification data from the user. This is implemented with forms that validate input according to certain criteria as it is entered; for example, ensuring that ids are the right length and passwords contain special characters.

4.1.1 Assumptions

- The user's device is connected to power or has enough battery life to last the duration of an operation.
- The user has an internet connection.
- The web application is running on the remote server.
- The user's technical abilities are sufficient to operate a login page.

4.1.2 Responsibilities

- Does not allow for usage or creation of Illegal accounts
- Stores the User's account data and privilege rank securely
- Allows for the creation of new accounts
- Allows for accounts to have their passwords reset

4.1.3 SUBSYSTEM INTERFACES

Table 2: Subsystem interfaces

ID	Description	Inputs	Outputs
#88132	A form that receives input	Input form	API endpoint
#78753	A form generated by the webapp	The webapp	Client browser

5 Dashboard Layer

The Dashboard refers to all interfaces through which the user may view or edit data stored by the application. Dashboard endpoints are generated as needed and take into consideration the privilege level of the user.

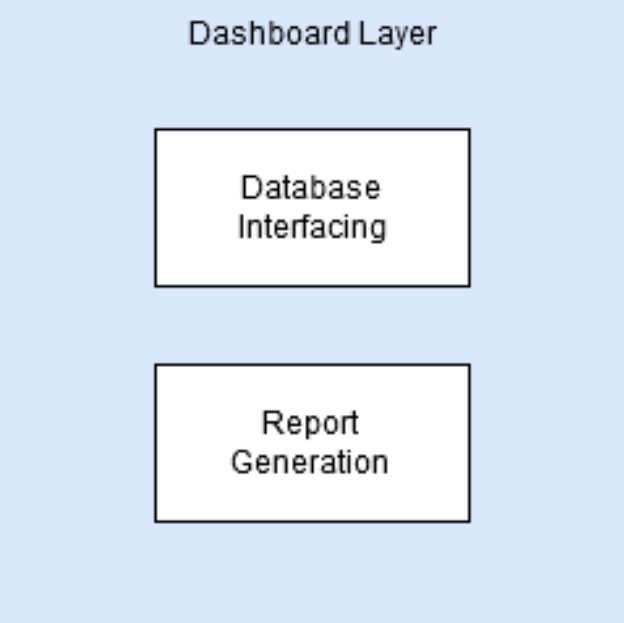

Figure 4: Dashboard subsystem diagram

5.1 Database Interfacing

Database interfaces serve two primary functions:

- presenting stored information to the user in an accessible format, and
- providing a secure method for editing/inserting into the database.

The combination and degree of functionality available to a user is dependent on his or her privilege level, as indicated by profile information.

5.1.1 Assumptions

- The user has an account stored in the database.
- The user has the appropriate privilege to perform the desired operation.

5.1.2 Responsibilities

If the user has *admin* privilege:

- Provide meaningful database views
- Securely handle modifications to the database

If the user has *student* privilege:

- Securely handle certain allowed modifications to the database

5.1.3 Subsystem Interfaces

Table 3: Subsystem interfaces

ID	Description	Inputs	Outputs
#36732	Database view	The webapp	Client browser
#92384	Form to insert/edit	Input form	API endpoint

5.2 Report Generation

The reporting component of the dashboard is tasked with making specific API calls to the web application to supply the user with PDF reports of inventory and checked out items. This function is only accessible to users with the *admin* privilege level.

5.2.1 Assumptions

- The user has an account stored in the database.
- The user has the *admin* privilege level.
- Desired information is actually present in the database.
- The user will provide a method for downloading/exporting the generated report file.

5.2.2 Responsibilities

The report generation module must make the appropriate API calls to the web application based on the output desired by the user. The unit must create a request and handle the server response, generating a file which can be downloaded or exported from the client browser.

5.2.3 Subsystem Interfaces

Table 4: Subsystem interfaces

ID	Description	Inputs	Outputs
#78611	Report generation form	Input form	API endpoint
#10097	Serving a file	Backend function	Client browser

6 Web Application Layer

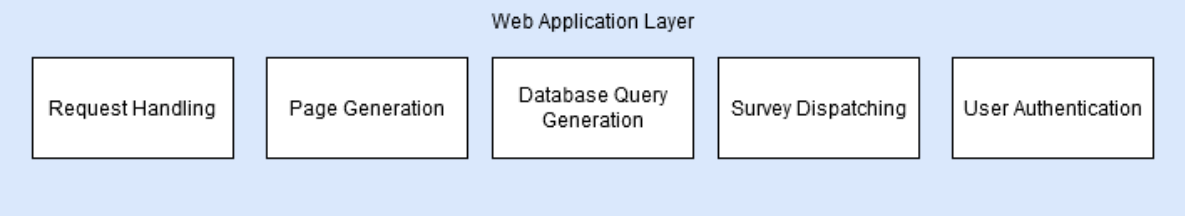

Figure 5: Web Application subsystem diagram

6.1 Request Handling

Primary function of this layer is to accept input from the user and send it the database.

6.1.1 Assumptions

- The back-end will verify input for malicious code.
- The user-interface will display the correct input forms.

6.1.2 Responsibilities

Making sure the input is sent to the correct API.

6.1.3 Subsystem Interfaces

Table 5: Subsystem interfaces

ID	Description	Inputs	Outputs
#83292	User Input	The web-app	POST request
#17793	Database	POST request	Confirmation

6.2 Page Generation

Primary function is to get the specified pages being requested for front-end.

6.2.1 Assumptions

- The page will be formatted to fit both mobile devices and computer screens.

6.2.2 Responsibilities

Generating the page whether it is a pdf form or a web page.

6.2.3 Subsystem Interfaces

Table 6: Subsystem interfaces

ID	Description	Inputs	Outputs
#10991	Get Page	URL	Page
#29820	Display Page	Page	The web-app

6.3 Database Query Generation

Primary function is to get information from the database to display in the front-end.

6.3.1 Assumptions

- The database will have backups in case of connection problems.

6.3.2 Responsibilities

Get information from database.

6.3.3 Subsystem Interfaces

Table 7: Subsystem interfaces

ID	Description	Inputs	Outputs
#10982	Database	Get Request	Information
#08963	Web-app	Information requested	Get request

6.4 Survey Dispatching

Primary function is to send survey that can update the information within the database.

6.4.1 Assumptions

- The survey can dynamically create new questions.
- Information gathered will be inserted within preexisting database models.

6.4.2 Responsibilities

Send survey to users in able to update or obtain information.

6.4.3 Subsystem Interfaces

Table 8: Subsystem interfaces

ID	Description	Inputs	Outputs
#86944	Survey Creator	Questions	Survey Form
#67688	Survey Form	User input	POST request

6.5 User Authentication

Primary function is to verify user credentials and account type.

6.5.1 Assumptions

- User cannot recover forgotten, can only reset to a new password
- User credentials can tie the User to their account
- User account types will not be NULL

6.5.2 Responsibilities

Confirm the users credential then confirm the account type to display the correct type of dashboard.

6.5.3 Subsystem Interfaces

Table 9: Subsystem interfaces

ID	Description	Inputs	Outputs
#40944	Login Page	User Credentials	User Credentials
#74921	Database	User Credentials	Dashboard Page

7 Database Layer

The database layer is a layer that no matter what privilege a user has, they should not be able to directly manipulate it. It is unique in the sense that while everyone can manipulate it using the forms defined in he webapp and Dashboard layer, direct manipulation will not be possible except for emergency situations, and then only by a database professional, not a common user.

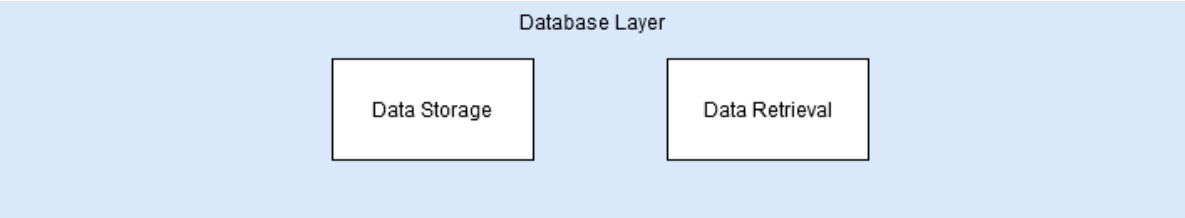

Figure 6: Database subsystem diagram

7.1 Data Storage

Databases are used to store data, so it makes sense that the most important part of a database is its ability to accurately store data. Our database will use inputs from our Dashboard that give the correct information into predefined fields, and will also be able to make new fields from the same Dashboard.

7.1.1 Assumptions

- correct credentials are provided
- consistent internet connection for communication
- no invalid insert calls

7.1.2 Responsibilities

- Store data without errors
- Create new fields based on new surveys
- Be able to store new data from Dashboard calls

7.1.3 Subsystem interfaces

Table 10: Subsystem interfaces

ID	Description	Inputs	Outputs
#1987.5	New survey adds new field	Webapp call	new data structure
#91324	User responds to survey	Webapp call	updated value
#42324	User updates equipment	Webapp call	updated value

7.2 Data Retrieval

Our database will need to be able to provide data to display to users on their browser, as well as to be able to generate a variety of forms with pre-populated data for printing.

7.2.1 Assumptions

- correct credentials are provided
- consistent internet connection for communication
- no requests for data that is not in the database

7.2.2 Responsibilities

- Provide requested data for UI to view
- Provide requested data for chosen form

7.2.3 Subsystem Interfaces

Table 11: Subsystem interfaces

ID	Description	Inputs	Outputs
#19875	Data is requested for UI view	Webapp call	visual data
#92792	Data is requested for form	webapp call	specific data

REFERENCES