

High Level Design Document

Version 1.0

Student Multi-Tool

12.15.2021

Team Marvel

Albert Toscano

Audrey Brio

Bradley Nickle

Joseph Cutri

Michael Kriesel (Team Leader)

Introduction	3
Scope	3
Overview	3
Design Details	3
Architecture	3
Presentation Tier	3
Business Tier	3
Data Tier	4
Abstraction Layers	4

Introduction

This high-level design document will give a comprehensive look at our solution from a general perspective and more mild technical terms.

Overview

This document will provide the following information:

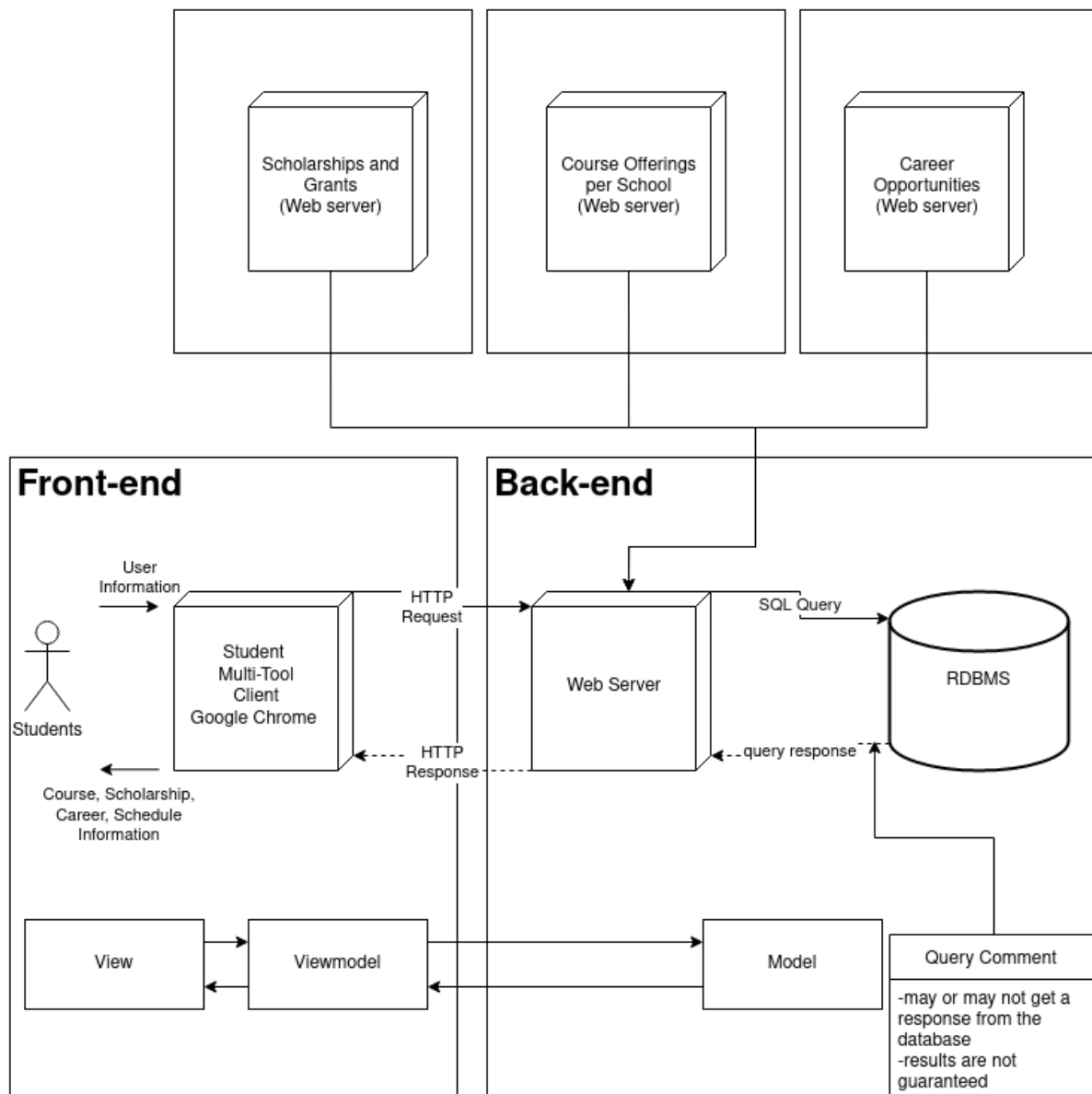
- Outlining our hardware architecture
- Our software architecture and how we utilize abstraction layers
- Our use of the MVVM pattern on a high-level

Hardware Architecture

Our product uses a 3-tier hardware architecture:

- Front end:
 - Client
- Back end:
 - Web Server
 - Communicates with other 3rd-party servers to retrieve external data
 - RDBMS

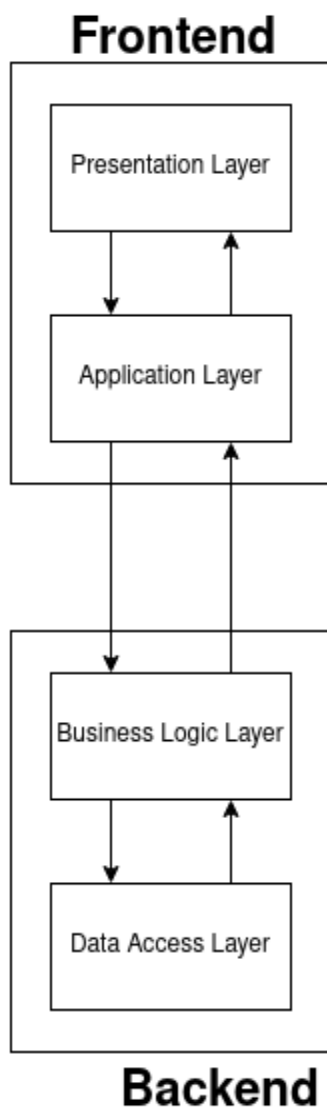
Hardware from various 3rd-Parties



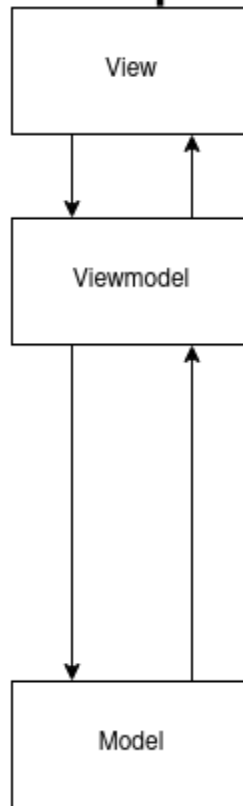
Software Architecture

Our product will use a 4-layer software architecture:

- Presentation Layer (front end)^[1]
 - Receives user input
 - Displays error messages
 - Sends and receives information to and from the application layer
- Application Layer (front end)^[1]
 - Performs input validation before sending data to the business layer
 - Performs client-side logic for schedule comparisons
- Business Layer (back end)^[1]
 - Performs server-side logic for:
 - Multiple-user features: collaborative schedule building, matching, messaging, etc.
 - Single-user features: user authentication, student discounts, career opportunities, aid eligibility
 - Retrieves information from 3rd-parties to send to the data layer:
 - Course offerings from CSU and UC schools
 - Scholarship and grant information
 - Career opportunities
 - Sends information from the application layer to the data layer to be saved
 - Performs error handling and logging
 - Performs user authentication
- Data Layer (back end)^[1]
 - Stores information and sends it to the business layer



MVVM Principles

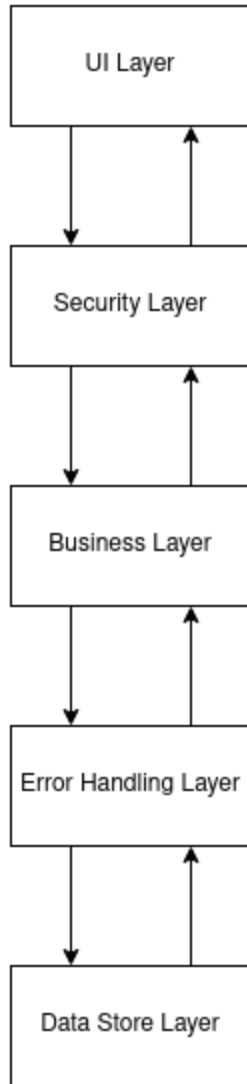


Abstraction Layers

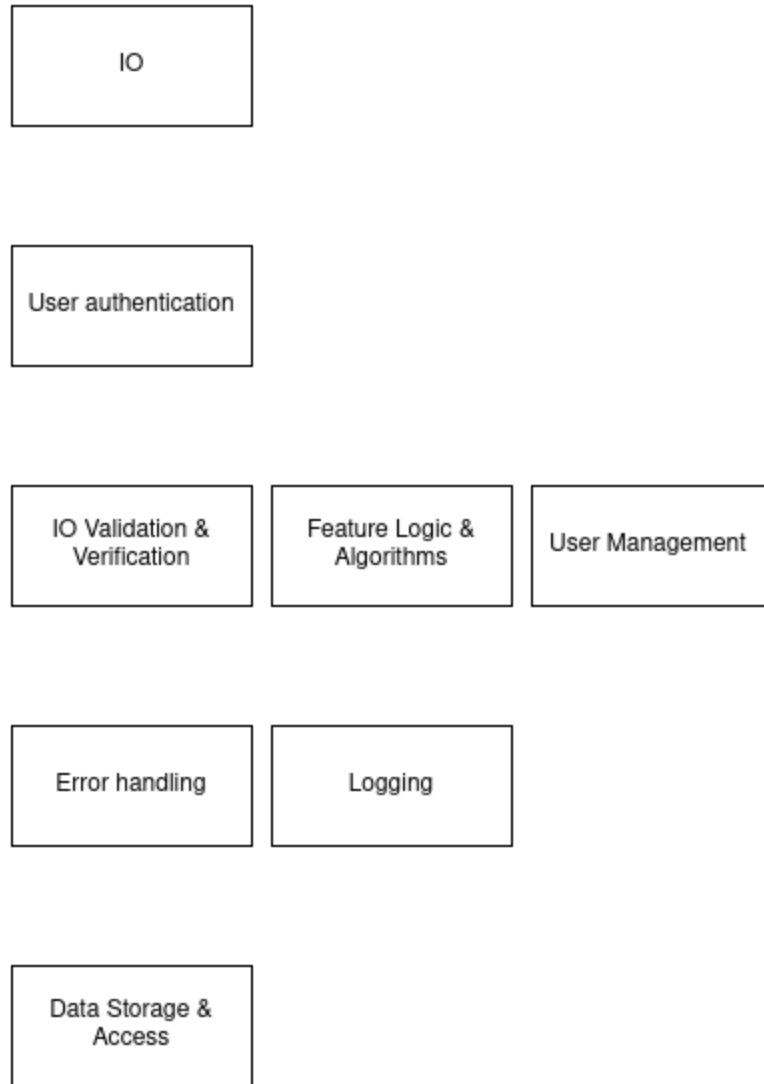
- User Interface Layer
 - Displays information for user in a variety of forms
 - Receives input from buttons and forms
- Security Layer
 - Authenticates user information
 - Authorizes users
 - User management performed here
 - Sign-in and sign-out functionality
- Business Layer
 - Handles the logical processing of data.
 - I.e valid characters
 - Determines matching schedules, carpool, and other activities.
 - Determines user's free time by comparing schedules
 - Determines difficulty of classes
 - Searches and posts student discounts
 - Determines user's aid eligibility estimation
 - Searches for student career opportunities
 - Displays the schedule of classes by semester
 - Sends and receives messages among users
 - Determines which information and actions are necessary to be logged, and in what manner
 - Implements search capabilities of logged information
 - Receives information from the internet
- Error Handling Layer
 - Processes error information to determine appropriate response.
 - Logs necessary error information
 - Sends system messages to user if necessary
 - Handles different levels of severity
 - **Critical (S1):** error affects critical data or functionality of the system.
 - Causes:
 - A complete failure of a feature
 - A blocked functionality
 - Actions:
 - Web browser displays a critical-error message
 - Web browser logs the error/data in the system
 - Web browser shuts down and nothing can proceed
 - Web browser is functional until the error is fixed
 - **Major (S2):** error affects major data or functionality of the system.
 - Causes:
 - A defect in the system
 - A feature is not functional
 - Actions:

- Web browser displays a major-error message
 - Web browser logs the error/data in the system
 - User is prompted to shut down the program to try to fix the error
 - Web browser is not fully functional until the error is fixed
- **Minor (S3):** error affects noncritical data or minor functionality of the system.
 - Causes:
 - Sometimes cosmetic errors (must be fixed)
 - A minor feature is not functional, but the system is functional
 - Actions:
 - Web browser displays a minor-error message
 - Web browser logs the error/data in the system
 - Other feature(s) can be used to perform the same task(s)
 - The system is functional, but features must be improved
- **Low (S4):** error does not affect data or functionality of the system
 - Causes:
 - Sometimes cosmetic errors (do not need to be fixed)
 - Spelling/grammatical errors
 - Enhancements in the existing design
 - Actions:
 - Design improvements
 - Grammatical corrections
 - The system is functional, but features can be improved
- Data Store Layer
 - Sends and retrieves data to and from the other layers
 - Gives each record the basic CRUD capabilities (Create, Read, Update, Delete) along with any other necessary features

Abstraction Layers



Responsibilities



References

1. https://en.wikipedia.org/wiki/Multitier_architecture, accessed 1 Oct. 2021