Adam Aguilar

Brandon Hohn

Reese Metcalf

Project – Week 2/3

Project Status Summary (1-page executive summary):

Current Progress Overview

The project continues on schedule with no major design issues or setbacks. The database structure for the CBU Tutoring Center Management System has been formalized, and initial MySQL scripts for table creation and population have been completed. Dummy data was successfully generated using AI tools, allowing the team to stress-test the schema and confirm referential integrity across entities. The data quality checks confirmed that the test set is clean and reliable, with both script-based and CSV-import methods available for flexibility in future migration or integration testing.

Sprint Retrospective Highlights

The team reports consistent progress and alignment with the original design plan. Collaboration through review ensured that table relationships, constraints, and key attributes were validated. AI-assisted data generation proved efficient, saving significant setup time while providing robust test coverage. The team also refined user stories, improving the precision interactions and acceptance criteria. These refinements clarified system behavior without requiring structural redesigns.

User Story Development

- **Story 1 – Tutor Updates Availability:** SQL logic was implemented to manage weekly availability limits and automatically refresh visible time slots.
- **Story 2 – Admin Adds New Tutors:** Initial logic and workflow design have been outlined for onboarding tutors, assigning subjects, and defining their availability.
- **Story 3 – Student Views Available Sessions:** The foundational framework for querying available sessions by class has been drafted and linked to the appointment and availability tables.

Technical Accomplishments

- Completed creation of all core tables: Student, Tutor, Subject, TutorSubject, Appointment, Availability, SessionFeedback, Room, and Staff.
- Validated referential integrity and behavior between foreign keys.
- Established prototype test scripts and queries to verify data flow between Student, Tutor, and Appointment modules.

- Developed stress-test datasets and verification methods for performance validation in future sprints.

Sprint Planning for Next Phase

The upcoming sprint will focus on integrating the data import mechanism, expanding user stories 2 and 3 into executable SQL logic, and building more complex queries to demonstrate functional workflows. Testing for time-slot conflicts, feedback analytics, and appointment tracking will begin. Jira tracking has been updated to reflect these priorities and assign ownership of key deliverables to each team member.

Summary Outlook

The project remains on track with all components established, and functional test data in place. The team's approach to validation, coupled with efficient use of AI for data generation, provides a strong foundation for upcoming feature testing and user interaction scenarios. The next sprint will transition the focus from structure validation to operational proof-of-concept testing and user experience changes.

Sprint Retrospective Meeting Summary:

As time has passed and work has continued, we have yet to hit any road bumps that would cause any major revisions; the ideas and designs thus far have remained solid. The team has decided on some graphical representations for the data that will comprise the tables as well as their attributes. MySQL Scripts have been made to begin creating the tables and preparing the data they will be using throughout the project.

With the tables taking shape and becoming more concrete, we decided to leverage AI for the use of generating a healthy amount of ammo for our growing database. Feeding the model with our specifications for the tables, we were able to easily generate an abundance of useful dummy data, which we then went through, vetting it for imperfections. Considering how quickly the data was generated, AI continues to show how promising it can be with certain aspects of project testing and development, like generating test cases and potential stress test scenarios. As we continue iterating through test cases and table specifications, it will be useful to have the AI environment ready to be able to adapt to new specifications we may want to quickly test.

The data and scripts generated so far appear to be clean and useful for feature testing. While generating and storing the data, we took two approaches; a file saved CSV to MySQL script approach and a singular insert script approach. Should we ever desire to

revert and migrate to a system that could utilize the system of pulling a file from a file's path on the hard drive, it will be nice to know we already have a small foundation there.

Alongside making the tables, data, and scripts, the user stories are progressing as expected and continuing to be worked alongside the other priorities. We didn't realize exactly how detailed and thorough each and every interaction stated within the User Stories needed to be while we initially wrote them. The good news is that the additional details don't change any major plans or general interactions planned for the actual database in the pipeline.

With the new data in place and the ability to rapidly develop stress tests nearly in place, we are eager to begin developing test cases to demonstrate our current design's strengths and potential weaknesses in the coming sprints.

Epic Summary:

Our group selected the CBU Tutoring Center Management System as the focus for our database project. This project is designed to support the Academic Success Center at California Baptist University by improving the way tutoring operations are organized and managed. The database will centralize data related to tutors, students, appointments, and session feedback, eliminating the need for manual tracking and allowing staff to efficiently coordinate tutoring services across departments.

The main goal of this project is to create a relational database that enhances the efficiency and accuracy of tutoring management. Currently, many academic centers rely on spreadsheets or manual sign-in sheets to handle scheduling, student-tutor matching, and feedback collection. Our system will automate these processes, making it easier for administrators to schedule sessions, match students to the appropriate tutors based on subject area, and monitor feedback trends. This will not only improve organization and data accuracy but also help the Academic Success Center identify areas for program improvement.

The system will include features such as tutor management to store and manage tutor profiles including their departments, subject specialties, and availability; student records to maintain student information such as academic departments/courses; appointment scheduling to create and track tutoring sessions while ensuring accurate scheduling and assignment of tutors; session feedback tracking to evaluate tutoring effectiveness.

Upon completion, the CBU Tutoring Center Management System will provide a streamlined and data-driven solution for managing the Academic Success Center's operations. The system will reduce manual work, improve data organization, and enhance decision-making through insights into tutoring effectiveness and student engagement. Ultimately, this project aims to strengthen the support for CBU students and contribute to a more efficient and responsive tutoring program.

User Story Summaries:

Story 1: Tutor Updating Their Availability

Persona:
Tutor

Story Summary:
A tutor needs to update their tutoring availability after a recent schedule change. The tutor wants to ensure that students can only see the correct, updated time slots when scheduling sessions. The system must enforce the tutoring center's 10-hour-per-week maximum availability policy and reflect these updates immediately so that scheduling conflicts are avoided.

Functional Description:
The tutor logs into the system and opens their schedule management interface. From there, they can view all their existing availability time slots. The tutor updates their schedule by removing old time slots and adding new ones that match their current availability.
The SQL logic below supports this process by first clearing out the tutor's old availability records, validating that the new total does not exceed the 10-hour weekly limit, and inserting the updated schedule into the database. This ensures that students only see the correct times when booking new appointments.

Use Case 1: Tutor Updates Their Availability

```
SET @TutorID = 8;

-- Example of new availability slots (must not exceed 10 hours total)

DROP TEMPORARY TABLE IF EXISTS TempNewSlots;

CREATE TEMPORARY TABLE TempNewSlots (

  DayOfWeek VARCHAR(10),

  StartTime TIME,
```

```
  EndTime TIME
);
-- Example: Tutor chooses 3 time slots (6 total hours)
INSERT INTO TempNewSlots (DayOfWeek, StartTime, EndTime) VALUES
  ('Monday',    '09:00:00', '11:00:00'),
  ('Wednesday', '13:00:00', '15:00:00'),
  ('Friday',    '10:00:00', '12:00:00');

-- Total Hours Entered
SELECT SUM(TIMESTAMPDIFF(HOUR, StartTime, EndTime)) INTO @NewHours
FROM TempNewSlots;
-- Check total available hours
SELECT IFNULL(SUM(TIMESTAMPDIFF(HOUR, StartTime, EndTime)),0) INTO
@ExistingHours
FROM Availability
WHERE TutorID = @TutorID AND IsAvailable = TRUE;
SET @TotalHours := @ExistingHours + @NewHours;
-- Hours Exceed Limit For The Week
SELECT
  CASE
    WHEN @TotalHours > 10 THEN 'ERROR: Cannot exceed 10 available hours per week.'
    ELSE 'OK'
  END AS ValidationResult;
-- Updates only if the Check is Passed
START TRANSACTION;
-- Remove old availability
```

```
DELETE FROM Availability WHERE TutorID = @TutorID;

-- Insert new availability (total ≤ 10 hours)

INSERT INTO Availability (TutorID, DayOfWeek, StartTime, EndTime, IsAvailable)

SELECT @TutorID, DayOfWeek, StartTime, EndTime, TRUE

FROM TempNewSlots;

COMMIT;

-- Verify new schedule

SELECT

  a.TutorID,

  CONCAT(t.FirstName, ' ', t.LastName) AS Tutor,

  a.DayOfWeek,

  a.StartTime,

  a.EndTime,

  TIMESTAMPDIFF(HOUR, a.StartTime, a.EndTime) AS SlotHours

FROM Availability a

JOIN Tutor t ON a.TutorID = t.TutorID

WHERE a.TutorID = @TutorID

ORDER BY FIELD(a.DayOfWeek,
'Monday','Tuesday','Wednesday','Thursday','Friday','Saturday','Sunday'),

    a.StartTime;
```

Story 2: Admin Adding New Users (BEGINNING IDEAS)

Persona:

Admin Authorized User

Story Summary:

An admin needs to add a new tutor, who was recently hired, to the system.

Functional Description:

The admin user should be able to add the tutors name, the subjects they teach, as well as their initial availability to their time slots.

Acceptance Criteria:

After receiving valid tutor information, once the new tutor is added, the new tutor user is shown in the director and is available to students who attempt to schedule with their available time slots.

Story 3: Student Viewing Available Time Slots For A Class (BEGINNING IDEAS)

Persona:

Student

Story Summary:

A student with a valid account, who is currently falling behind in a subject, wants to view the current available tutoring time slots for their class.

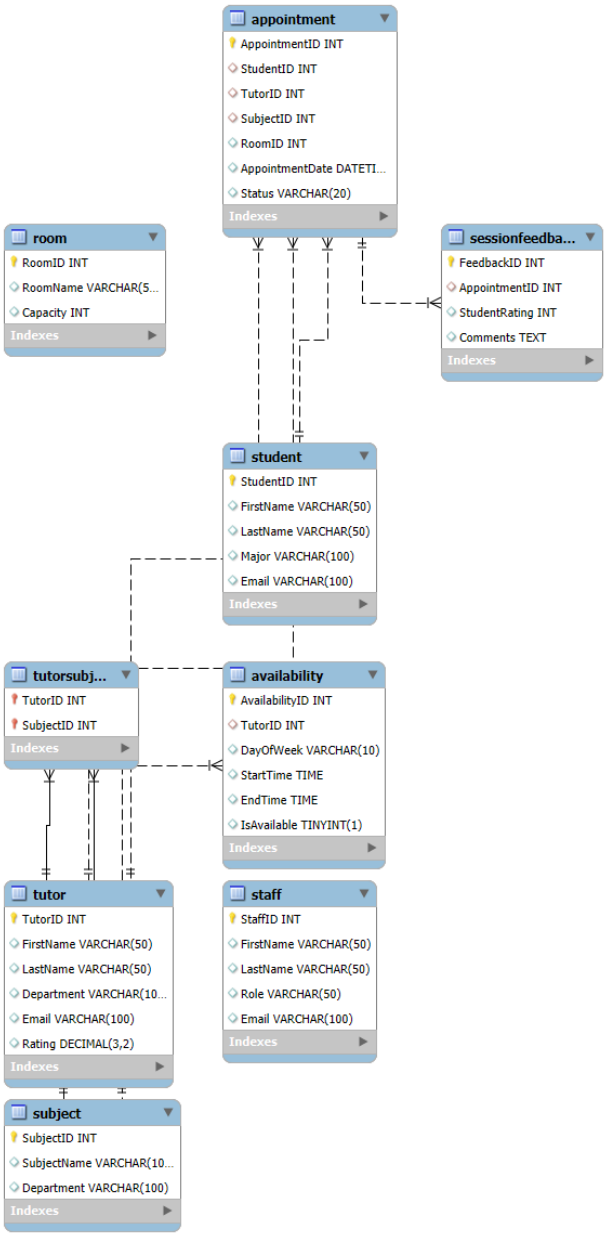
Functional Description:

The student with a valid account is able to log in, select a class, and the relevant time slots are shown to them.

Acceptance Criteria:

When the student is logged in, with a valid account, and queries using valid course information, is given the list of available time slots by the system.

Graphical Representation of the Logical Model:



All SQP Script Developed thus far:

```
#CREATE DATABASE CBU_TutoringCenter;

#USE CBU_TutoringCenter;


CREATE TABLE Student (
    StudentID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
```

```
    LastName VARCHAR(50),
    Major VARCHAR(100),
    Email VARCHAR(100) UNIQUE
);

CREATE TABLE Tutor (
    TutorID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Department VARCHAR(100),
    Email VARCHAR(100) UNIQUE,
    Rating DECIMAL(3,2)
);

CREATE TABLE Subject (
    SubjectID INT PRIMARY KEY AUTO_INCREMENT,
    SubjectName VARCHAR(100),
    Department VARCHAR(100)
);

CREATE TABLE TutorSubject (
    TutorID INT,
    SubjectID INT,
    PRIMARY KEY (TutorID, SubjectID),
    FOREIGN KEY (TutorID) REFERENCES Tutor(TutorID),
    FOREIGN KEY (SubjectID) REFERENCES Subject(SubjectID)
```

```
);

CREATE TABLE Appointment (
   AppointmentID INT PRIMARY KEY AUTO_INCREMENT,
   StudentID INT,
   TutorID INT,
   SubjectID INT,
   RoomID INT,
   AppointmentDate DATETIME,
   Status VARCHAR(20),
   FOREIGN KEY (StudentID) REFERENCES Student(StudentID),
   FOREIGN KEY (TutorID) REFERENCES Tutor(TutorID),
   FOREIGN KEY (SubjectID) REFERENCES Subject(SubjectID)
);

CREATE TABLE Availability (
   AvailabilityID INT PRIMARY KEY AUTO_INCREMENT,
   TutorID INT,
   DayOfWeek VARCHAR(10),
   StartTime TIME,
   EndTime TIME,
   IsAvailable BOOLEAN,
   FOREIGN KEY (TutorID) REFERENCES Tutor(TutorID)
);

CREATE TABLE SessionFeedback (
```

```
    FeedbackID INT PRIMARY KEY AUTO_INCREMENT,
    AppointmentID INT,
    StudentRating INT CHECK (StudentRating BETWEEN 1 AND 5),
    Comments TEXT,
    FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
);

CREATE TABLE Room (
    RoomID INT PRIMARY KEY AUTO_INCREMENT,
    RoomName VARCHAR(50),
    Capacity INT
);

CREATE TABLE Staff (
    StaffID INT PRIMARY KEY AUTO_INCREMENT,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Role VARCHAR(50),
    Email VARCHAR(100)
);
```

Sprint Planning Meeting Summary:

Week 2:

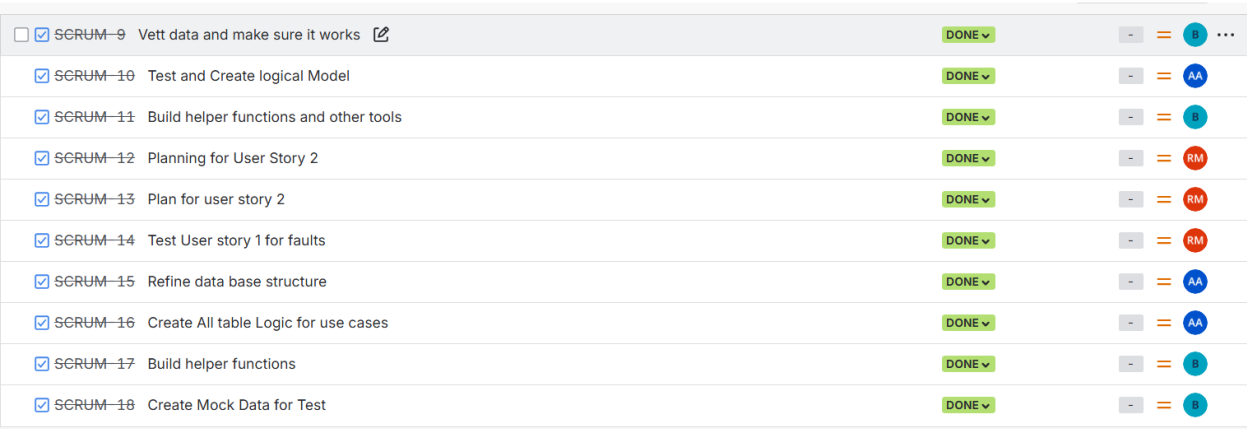
- Begin generating dummy data and helper functions to test the data with the current database model (Brandon)
- Apply all test data into tables (Adam)
- Test data flow between all tables for structure viability (Reese)

- Begin building queries for base user story cases (Reese)
- Reevaluating database structure after testing (Brandon)
- Check to see if customer needs are met for base cases (Adam)

Week 3:

- Experiment with making a system that allows for importing data from a file path instead of purely the insertion script (Brandon)
- Setup work for user stories 1, 2, & 3 (Adam/Brandon)
- User story write up (Reese)
- Finish structure for user story 1 and early tests (Reese)
- Plan User stories 2 and 3 for design (Reese)

Screenshot of Jira page for sprint that was completed:



Screenshot of Jira page for upcoming sprint: