**Compiled Documentation of all Deliverables:**

**Helping Hands: System Request**

University of Maryland, Baltimore County

Professor: Shiva Sharma

IS436:1645

**Date:** 02/20/2019

**Team Members:**

Timnit Tesfatsion

Daniel Bae

Briana Jenkins

Wonjin(James) Choi

Anusha Malla

Haythem Abdelkhalek

**------Deliverable 1: System Request -----------**

|  |
| --- |
| **System Request** |
| **Project Title:** Helping Hands  **Project Sponsor:** Ira Fabri, Supervisor of the Tutoring Program at UMBC  **Date of Authorization:** February 14th by Professor Sharma  **Project Start Date:** February 13th **Project Finish Date:** TBD |
| **Key Schedule Milestones:**   * System Request and Feasibility (10%) 1 month * Planning (15%) 1 month * Analysis (25%) 2 months * Design of App (25%) 3 months * Implementation (25%) 3 months |
| **Budget Information/Value:**  We expect Helping Hands to:   * increase profit and increase consumer productivity by providing college students the necessary tools and resources, specifically tutors, that will enable them to be successful in their classes. * elevate the number of subjects our tutors are able to offer in order to expand our clientele. * provide better ways of communication from student to scheduled tutors by providing a stream of notifications and using internal app’s chat system as a conduit for communication.   Company estimated value include the following:   * $8,000 in sales per semester per university. |
| **Project Manager:** Timnit Tesfatsion, tt5@umbc.edu |
| **Project Objectives/Business Need:**   * This app will help UMBC save money by making its own app.   + Long-term, after debug and implementation, UMBC could sell this product to other universities.   + The ultimate goal is that this app would make any university’s tutoring service to be independent. The supervisor would have full control of the app (but not the code). |
| **Main Project Sources Criteria:**  Task 1:   * Give permission to students and tutors the ability to change their schedules. For high-demand, you can use the app to schedule a session. This means that your schedule has to be correct and up-to-date. * For low-demand, students have to contact the supervisor of the Learning Resource Center through email to be able to see if there’s any tutors for the course they need help with.)   Task 2:   * All courses have to be listed on the app. No emails between students and tutors.   Task 3:   * The app needs to have a simple messaging system for tutors and students to chat (needed when a student is not coming or going to be late, or simply to ask any information).   Task 4:   * Supervisors can give permission to new hired tutors to become a “tutor” in the app.(Right now, supervisors have to contact StudyTree support to ask for it. This usually takes days).   Task 5:   * Students can add personal details in their profile, like “student with disability”, or “retaking this course”, or “Need to practice speaking only (for language tutoring)”. This will help tutors prepare for their sessions before hand and be more efficient. |
| **Approach/Requirements:**   * We are going to use the android studio software * Using SQL database with java for the coding aspect |
| **Special Issues or Constraints:**   * There are multiple apps on the market that provide tutoring for students. * In order to gain a big pool of clients, we must complete this project as soon as possible. We would like this app to become mainstream and have students default to our app instead of using other apps that may provide similar resources. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Roles and Responsibilities** | | | |
| **Name** | **Role Description** | **Bio** | **Contact Information** |
| Ira Fabri | Project Sponsor | Supervisor of the Tutoring Program at UMBC | [fabri@umbc.edu](mailto:fabri@umbc.edu), (410) 455 3905 |
| Timnit Tesfatsion | Project Manager: I will overlook the overall project from start to finish. Also, I will coordinate meeting times as well as work with each team member to meet deadlines as well as assign responsibilities. | I am an information Systems major and I have some knowledge with SQL and I have experience with project management based on previous class I took. | [tt5@umbc.edu](mailto:tt5@umbc.edu), (301) 747-8721 |
| Daniel Bae | Programmer: As the lead programmer, I will oversee and code the basic structure of our app. It will encompass Java, HTML, and SQL code with the possibility of additional languages. | I have a bachelors in Biological Sciences; I am 24 years old | dbae1@umbc.edu , (443) 845-8434 |
| Briana Jenkins | System Design Analyst: takes both jobs of a system analyst and a computer programmer. I will be designing the design layout for the app. I will be using the android studio software to complete this task. | My name is Briana Jenkins. I am 23 years old, and a senior. | bri20@umbc.edu , (301) 609 -1765 |
| Wonjin (James) Choi | QA/Business Systems Analyst: Ensure software is implemented with the requirements needed. Conduct analysis to create long term business solution. | I’m a 24 year old senior majoring in information systems. | choiw1@umbc.edu , (443) 798-5540 |
| Anusha Malla | QA: I will be manually testing the app and the codes to prevent malfunctioning code or app. | I come from a landlocked country Nepal. I am 21 years old, can speak 4 different languages and a senior in Information Systems. | vc15539@umbc.edu , (443) 722-9151 |
| Haythem Abdelkhalek | Database Developer: I will design a fully functional SQL-PL/SQL database with functions, procedures, and triggers that will be the base of the App. | Senior Information System’s major working part-time as a database developer. | haythem1@umbc.edu , ( 410) 599-3451 |

|  |
| --- |
| **Group Meeting Times** |
| * Every Wednesday from 7pm to 8pm in the RLC   + If school is closed due to weather condition, our group meets online via Google Hangout * Throughout the week, we chat through GroupMe for updates, reminders ...etc |

**Feasibility Analysis**

|  |
| --- |
| **Helping Hands Executive Summary**  Our group have made this feasibility analysis for Helping Hands. |
| **Technical Feasibility:**  What kind of Risks (low, medium, high)   * Familiarity   + Medium to high risk since only one member of the team worked on apps before.   + We have a member that have worked on GUI development at a university course, in order to implement an app.   + Low risk for familiarity since the current system is being used. * Project size   + Will take 3-5 months   + Low risk because it’ll be slowly implemented. Not a massive project.   + Medium to high risk with how the UMBC system will integrate with the tutoring as well as managing the system. * Compatibility   + Medium to High Risk : Have to rely on data by manual implementations… StudyTree may not provide existing data, but the data is somewhat public. Once data is inserted the system will work fine.   + Low Risk since system can run by itself when data is implemented. |
| **Economic Feasibility:**  Based on cost-benefit analysis calculations (see Cost-benefit Tables below). The estimations shows that Helping Hands App will increase UMBC’s performance in its tutoring system.   * ROI over 12 months: 70% * NPV over 12 months: $6991 * Break-even occurs after 1.26 years   What are the economical benefits of our app?   * Elevated customer satisfaction which would increase application purchase * Lower cost maintaining local app * Increase in the number of students who may not know about this service, potentially increasing revenue. * Chance for competition and expansion of app to other universities * Create jobs for students and other staff members |
| **Organizational Feasibility:**  What kind of risk does this app pose to umbc?   * It can pose a low risk if the supervisor is liking the idea and thinks it can be beneficial * It can pose as high risk if they think they already have an alternative option and thinks it will be a little use to them * Servers going down or not working properly * Another low risk is having the issue that if a student does not show up the tutor still get paid so in that case on the economic part of it, it will cause money loss   Does the supervisor of our project have interest? What kind? Expectations?   * Supervisor needs faster support for schedule changes. * They need instant ability to add new tutors to the system.   What kind of reaction/feedback are we expecting from the feature user of our app? What about the concerns they might have? Challenges/competitions?   * That it would be easier for the students to operate when wanting to schedule or reschedule for tutoring * It would become easier for the students to be able to cancel, update or stay in contact with tutors * Having the ability to use the app easily * Make it easier for them so that they can talk to friends about the app and then we can get many more students to use it more often |
| **Additional Comments:**  How do we (group members) see this project?   * This project has a profitable app because all students will have access to it * Opportunity for students to create a service that enables other students to do better in specific courses that are difficult   What are our expectations of doing this app?   * Increase student awareness on a potential service that could potentially increase a students grade. * Individual and unique learning experience. * Easier scheduling process for all participating parties   Do we think our app will be well accepted?   * It will be accepted because all students need a helping hand and the app helps to get to that point * The app helps with finding a tutor faster and easily * This app will help in time flexibility   Any challenges we might face whether technical or anything else?   * The coding might be a little tricky * Getting all of the universities to buy the app might take some time |

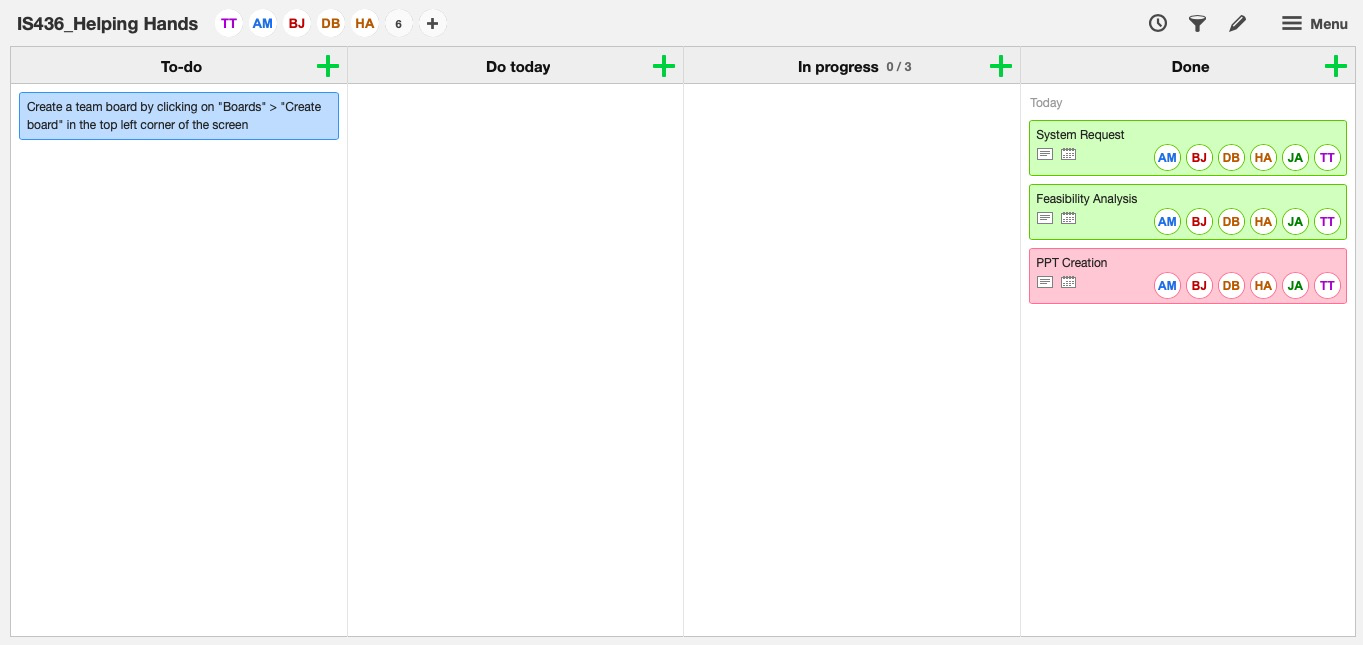
**Cost-Benefit Analysis - Simple Cash Flow Method**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 4 months | 8 months | 12 months | Total |
| **Benefits**   * Increase in the number of users * Reduction in number of phone-calls for support from supervisors * Reduced number of emails between   + Supervisor and developers   + Tutors and supervisor   + Students and supervisor |  | 5800  300  1480 | 6960  360  1776 | 12760  660  3256 |
| **Total Benefits** |  | 7580 | 9096 | 16676 |
| **Development Cost**   * Server * Software licences * Server software | 100  100  100 | 0  0  0 | 0  0  0 | 100  100  100 |
| **Total Development Cost** | 300 | 0 | 0 | 300 |
| **Operational Cost**   * Software update/maintenance * Communication charges * Operational charges |  | 50  200  4000 | 80  400  4500 | 130  600  8500 |
| **Total Operational Costs** |  | 4250 | 4980 | 9230 |
| **Total Costs** | 300 | 4250 | 4980 | 9530 |
| **Total Benefits - Total Costs** | (300) | 3330 | 4116 | 7146 |
| **Cumulative Net Cash Flow** | (300) | (200) | 7146 |  |
| **Return on Investment (ROI)** | 70%  (7146/ 9530) |  |  |  |
| **Break-even Point** | 1.26 years |  |  |  |

**Cost-Benefit Analysis - Discount Cash Flow Method:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 4 months | 8 months | 12 months | Total |
| **Benefits**   * Increase in the number of users * Reduction in number of phone-calls for support from supervisors * Reduced number of emails between   + Supervisor and developers   + Tutors and supervisor   + Students and supervisor |  | 5800  300  1480 | 6960  360  1776 |  |
| **Total Benefits** |  | 7580 | 9096 |  |
| **Present Value Total Benefits** |  | 7343 | 8536 | 15879 |
| **Development Cost**   * Server * Software licences * Server software | 100  100  100 | 0  0  0 | 0  0  0 |  |
| **Total Development Cost** | 300 | 0 | 0 |  |
| **Operational Cost**   * Software update/maintenance * Communication charges * Operational charges |  | 50  200  4000 | 80  400  4500 | 130  600  8500 |
| **Total Operational Costs** |  | 4250 | 4980 | 9230 |
| **Total Costs** | 300 | 4250 | 4980 | 9530 |
| **Present Value Total Cost** | 300 | 4117 | 4671 | 9088 |
| **NPV (PV Total Benefits - PV Total Costs)** |  |  |  | 6791 |

**KanbanFlow:**



**Deliverable 2:Requirements Definition Document and Use Cases**

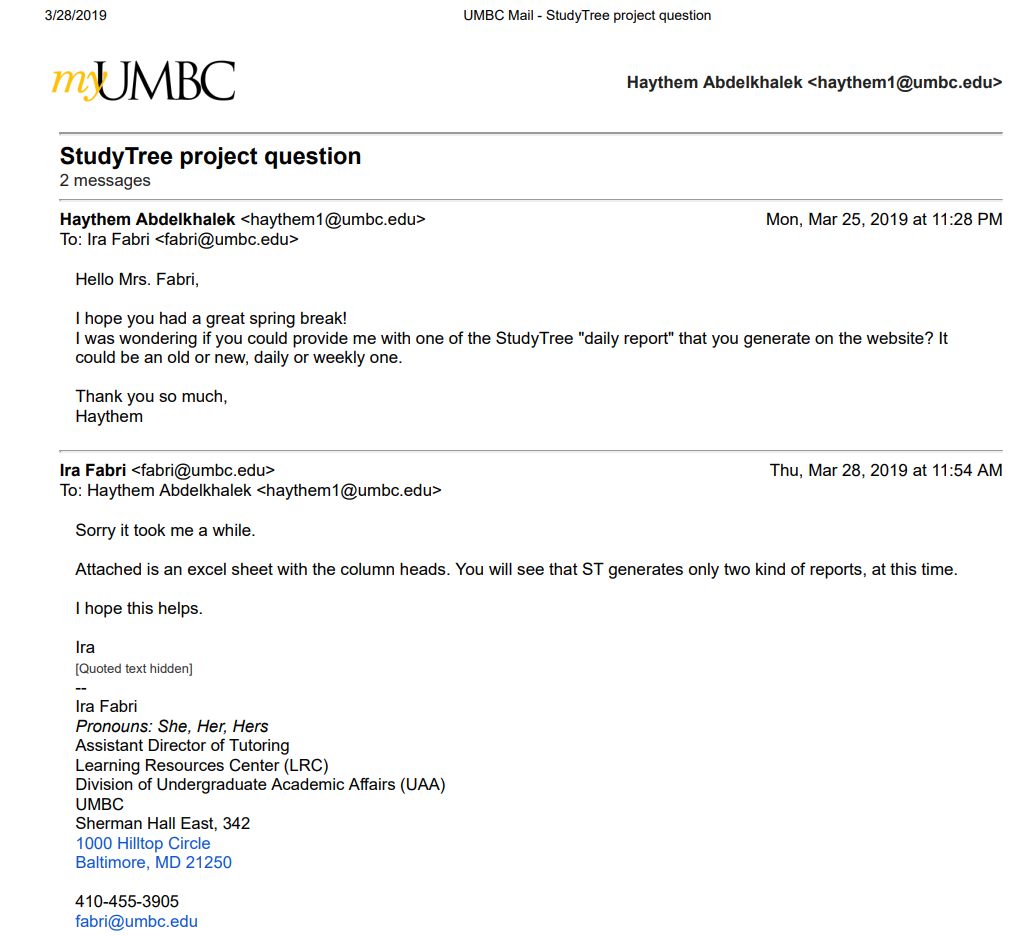
**Interviews**

|  |
| --- |
| **Interview Notes Approved by: Ira Fabri** |
| **Person Interviewed: Ira Fabri**  ***Assistant Director of Tutoring***  **Interviewer: Haythem, Briana, James**  **Date/Time: 03/15/2019 at 2:30pm** |
| **Purpose of Interview:**   * Gain an understanding of how tutoring sessions are being scheduled and tracked. * Determine what is required for the future system. |
| **Summary of Interview:**   * Tutoring sessions are currently being made through a third party app called StudyTree and this slows down support when it’s needed. * Ira is responsible of scheduling weekly appointment, contacting StudyTree support whenever it’s needed, and coordinate between tutors and students when schedules do not match. * Ira reports to Cassie with weekly and monthly StudyTree generated reports. * The new systems needs to allow tutors to make changes to their own schedules. |
| **Open Items:**   * Get a daily report from Ira. * Get a copy of Ira’s tutors evaluation form. |
| **Detailed Notes:**  Q: What is your role in the supervisor position?  A: I manage tutoring sessions, this means adding, scheduling, canceling, and editing sessions. I also work very closely with StudyStree support team as I am responsible of fixing issues with the app that students or tutors encounter. I also evaluate current tutors and hire new tutors every semester.    Q: How many students tutors teach daily?  A: Right now, average daily students is around 8 students, but some days like Wednesdays, we get around 20 students, and Fridays around 2 students.  Q:How many tutors do you roughly have?  A: For this semester 73 and we usually never exceed 80.  Q: How does the current system make scheduling a student with a tutor convenient?  A: For an administrator, I simply login to the StudyTree website and create a session by choosing between the different options that match with the student needs. From a student perspective, the app is linked to Blackboard so they only see tutoring for their current courses  Q: What is your opinion on the study app?  A: The options it offers are excellent, but it has too many issues and bugs that makes it really difficult to use.  Q: How can the app improve its scheduling system to make it easier for students?  A: From what I heard from students, the whole concept of AI that helps you pick up available tutoring sessions is very confusing. Students like it when it’s simple and clear. It would be easier if students can log into the apps and see a list of the available sessions in the course they need help with right away. Students should also be able to choose if they want to make their appointment recurrent weekly or just once at a time. This will eliminate a lot of unnecessary emails exchanges between me and the students. We also need to create waitlists for sessions that are already booked. This way, students don’t have to constantly track if their favorite time for a session is available or not.  Q: How can the app improve its scheduling system to make it easier for tutors?  A: Tutors need to be able to input their own availability for tutoring on the app and change it at any time they need. Currently, all tutors send me their schedules and I have to enter them manually into the system. In addition to that, If a tutor is going to be absent for being sick or any other reason, they need to send me an email to ask me to cancel their sessions and then I have to send emails to the students to inform them that the session is cancelled and the reason of cancellation.  Q: How often do you get data reports and how can it be improved?  A: I can request reports anytime, but those reports are always bugged and it takes around 15 min to get one. For example: if a student cancel a weekly appointment, the report will keep showing 13 cancellations for every single week that session was supposed to happen. This makes it very difficult to interpret in Excel and visualize. I need simple spreadsheet with more accurate numbers that shows the difference between weekly sessions and one time sessions.  Q: Is there a feature on StudyTree that help you evaluate tutors? How do you currently evaluate students?  A: No. Right now, I have to assign tutors to evaluate their peers in their down time. They answer a questionnaire I shared with all tutors as a Word document. Every two week as I have to read those evaluations to make sure everything is going well, especially for new tutors.  Q: Would it help if the system included a tutor evaluation feature? How would this feature work?  A: Yes! I would like to have my questionnaire adapted and made available for tutors on the app. Instead of waiting for the evaluators to finish answering the questionnaire and sending it to me, I need to have a score of how well is the tutor being evaluated is doing as soon as session ends.  Q: How can the system help tutors prepare for their tutoring session?  A: Students need the ability to add information to their profile. For example: we need to know if a student needs special accomodation and provide it to them if needed.  Q: Does the system track No-shows?  A: No. This is an important feature we need to add to the current system in order to block students from booking sessions after 2 consecutive No-shows. |

|  |
| --- |
| **Interview Notes Approved by: Cassie** |
| **Person Interviewed: Cassie Hoddinott**  **LRC Director**  **Interviewer: Haythem, Briana, James**  **Date/Time: 3/16/19 at 3:30pm** |
| **Purpose of Interview:**   * Understand Cassie’s role in the tutoring center and how she interacts with the current system. * Determine what is required for the future system. |
| **Summary of Interview:**   * Weekly and monthly reports are crucial for the LRC to receive support from administration. * Administration should be able to assign tutoring classrooms to tutors. * Students should be able to create their profile to help tutors prepare for their sessions. * Direct messaging is needed between support and students. |
| **Open Items:**   * Get a weekly report from Cassie * Get the features analysis |
| **Detailed Notes:**  *Q:* What is your role in the supervisor position?  *A:* I am the director of the LRC. This means I get weekly reports from all the staff and analyze those reports to make strategic decisions.    *Q:* How many staff members can view and contribute to the StudyTree reports?  *A: Currently it is just Ira and I.*  *Q*: How is your role within StudyTree different than Irs’s?  A: Ira is the person who is in direct contact with the tutors, students, and StudyTree support. I am the one who fixes the number of tutors we can hire, the maximum number of sessions students can get, and answer to all the financial matters between LRC and UMBC.  Q: What makes using StudyTree to schedule tutoring convenient?  A: We started using StudyTree because it is cheap and it offers good data analytics and daily/weekly/monthly reports that help us make decisions. Before using this app, we used to make tutors and students fill out Google Forms which limit us in the amount of data we could collect about tutoring sessions.  Q: What is your opinion on the StudyTree app?  A: StudyTree was not programmed to work with a tutoring center like the LRC. It was mainly designed to put together tutors and students without going through any staff members which is impossible to do as any organization or University need to supervise and manage their tutors constantly.  Q: What are some improvements that could be made for the app?  A: First of all, there are too many issues and bugs in the app that makes it crash every 10 min and this needs to be fixed. Tutors need to be able to change their available times to tutor. Administrators should be allowed to assign classrooms to the session when they created. Right now, all session are listed to happen in “UMBC” but no other details. Students should be able to create a profile in order to add specific information about what they need help with, or if they need accomodations. Tutors should be able to create a profile to select what courses they want to mainly tutor and which ones they can potentially tutor.  Q: Is there a way communication between administration and students can become more efficient?  Q: Students should be able to ask for support directly on the app. Ira and I get around 50 emails every week from students and we have to transfer those email to support so they can solve their problems. This is extra work when we should be focus on more important tasks. |

**Document Analysis**

From the interviews that we have conducted, we have asked the representatives of the tutoring center to prepare us document(*Scheduling system specs*) that display the requirements they need in the app we will be building. For instance, what they are looking to be implemented for tutors, students and supervisors. The representative have also shared an Excel sheet *(ASAC Scheduling Platform Comparison*) that compared different Scheduling Platform.

****

For Privacy issues, we were not allowed to share any reports containing data.

For the purpose of this deliverable, below is a list of the column names in the 2 possible reports. These reports are generated in a .xlsx document

**Tutoring Center Usage report**

Tutoring Center Name - Resource Name - Course - Session Id - PoolSession Id - Tutor First Name - Tutor Last Name - Tutor Email - Tutor SIS Id - Student First Name - Student Last Name - Student Email - Student SIS Id - Creator First Name - Creator Last Name - Creation Time - Scheduled Start Time - Scheduled End Time - Event Status - Event Deleted By - Event Deletion Reason - Event Deletion Time - Is No Show

**Confirmed sessions report**

Session Id - PoolSession Id - Resource Name - Course - Tutor First Name - Tutor Last Name - Tutor Email - Tutor Average Rating - Student First Name - Student Last Name - Student Email - Start Time - End Time - Session Duration - Session Review Score By Student - Session Review Score By Tutor

Current improvements and features requested by Ira from StudyTree to implement in the next update.

**Scheduling system specs**

Required

1. **Create as many schedules as necessary**
2. **Decide number of allowed appointments per week**
3. **Allow different start/end times and lengths of appointments**.
4. **Schedule accessible from multiple devices/locations**
5. **Students can search by topic**
6. **Forms (appointments, report, registration, etc.) customized easily**
7. **Waiting List**
8. **Send appt. confirmations/reminders, different times and formats (email or text or both)**
9. **Can create tutor profiles**
10. **Survey option**
11. **AU Email registration only**
12. **Track no-shows and block offenders**
13. **Generate reports and analytics**
14. **Good customer support**

Strongly preferred

1. Forms (appointments, report, registration, etc.) customizable differently for different centers.
2. Sync appointments with other calendars
3. Opening/closing hours is easy for tutors to do on their own
4. Online session capability

Preferred but not indispensable

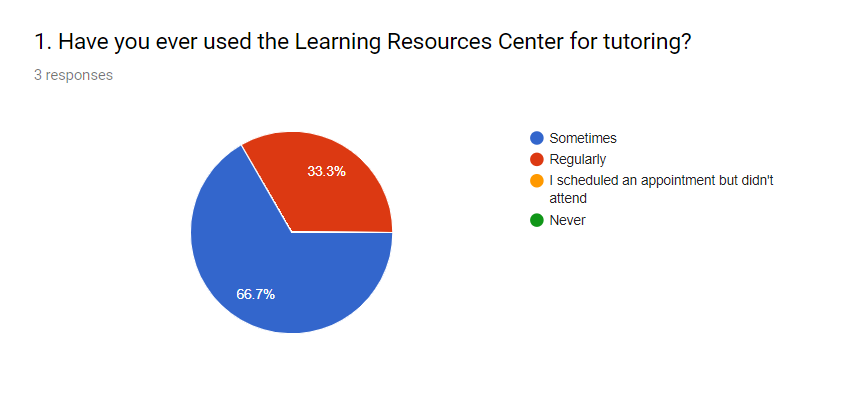
1. Tracks sign ins/outs by student's number (good for SI and drop-ins)
2. Faculty referral and early alert system
3. Offers their own online tutors
4. Online chat/messaging function or text feature
5. Facebook integration
6. Tutor payroll
7. Interface with SIS

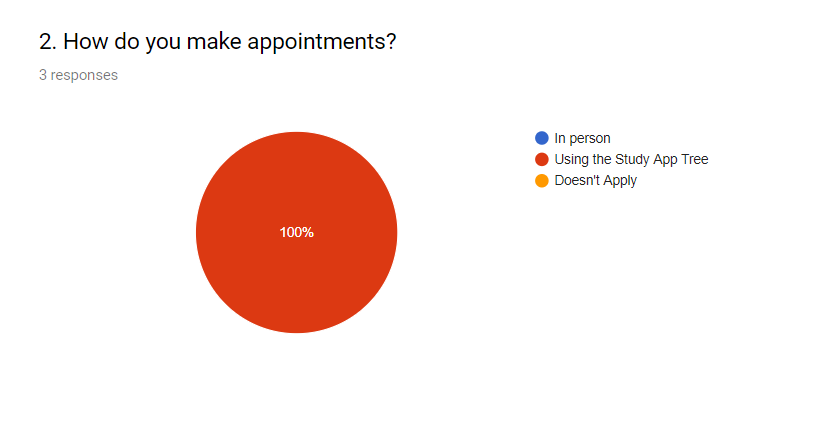
**Observation**

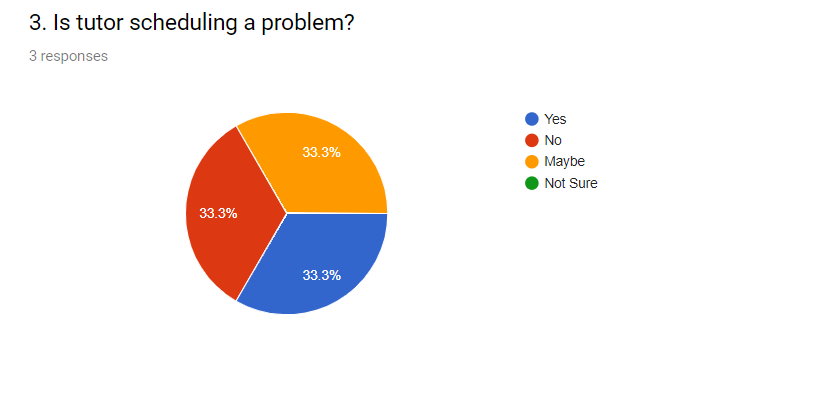
We started the observation by logging into Ira’s administrator account on StudyTree’s website and going through today’s list of sessions, March 15, to check the accuracy of the data before generating a StudyTree daily report. Ira pointed out that the number of sessions displayed in the schedule does not reflect the real number of session happening today. Few sessions were missing and few others showed a start time at 9:00am which is an hour before the Learning Resources Center opens. Then, we checked her inbox to check the nature of email exchanges she gets from tutors and students. At 3:15PM, just for that day, she had received 5 emails from tutors and 10 emails from students. It seemed that these emails had many things in common. Tutors would be asking about making changes to their schedules, as they are unable to do so with the current system and Ira has to do it manually, and students requesting weekly sessions instead of having to manually book sessions every week. Then we generated a daily StudyTree report to examine the data gathered for the tutoring center. Generating the report took about twelve minutes of loading time. The report showed the correct sessions scheduled with correct status “No-Show, completed, not started” but did not flag the repeated no-shows to block students with more than two absences in a row from booking new sessions. The report also showed thirteen cancellations for two students who cancelled their weekly appointment. From Ira’s perspective, this should show-up only once and a new column with the number of sessions being cancelled in a new column. Ira explained how having well accurate and well organized report is crucial as she uses them to show the Learning Resources Centers’ activities to UMBC administration.

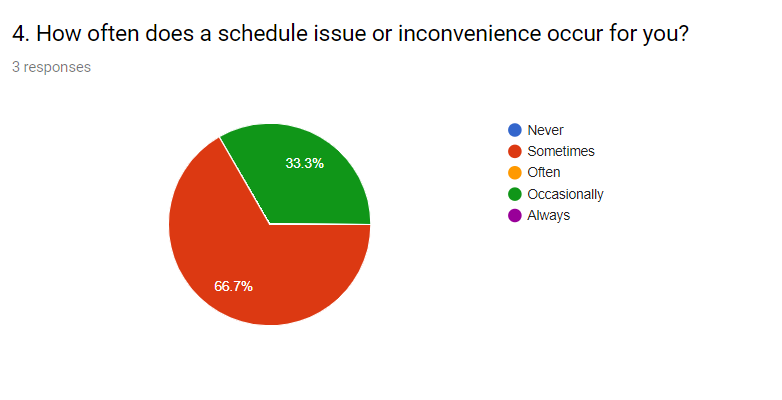
**Questionnaire**

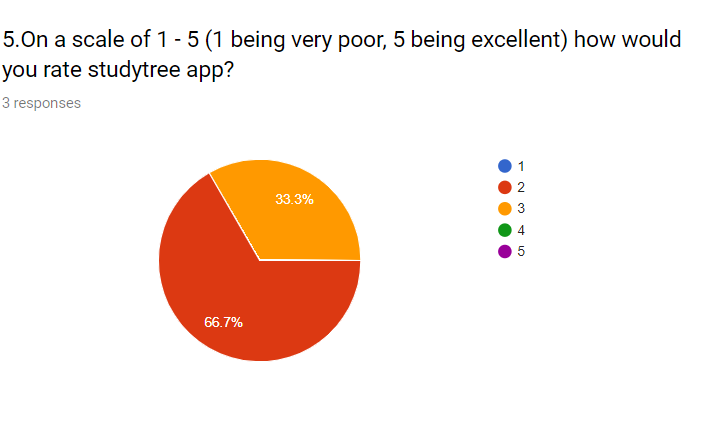
Our Questionnaire was created with Google Forms. Our group member Haythem Abdelkhalek works currently at the Learning Resources Center as a tutor. Haythem sent the questionnaire to all his current students and to other tutors so they can forward it to their own students. The questionnaire was sent to students on Sunday March 24th and responses were collected until March 27th. The Google Form was sent out to 16 staff and students, only 3 responded.

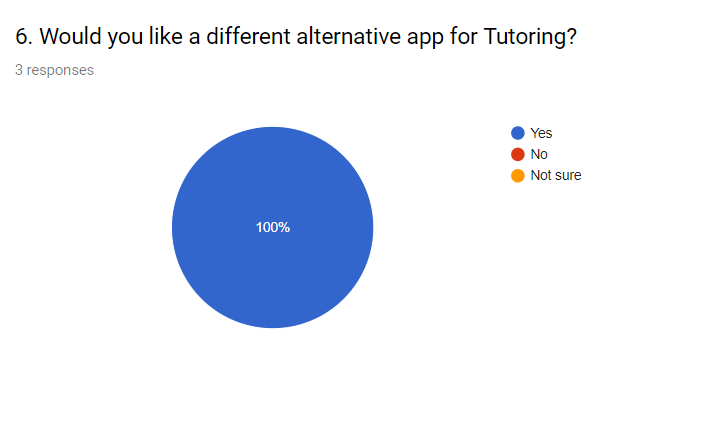


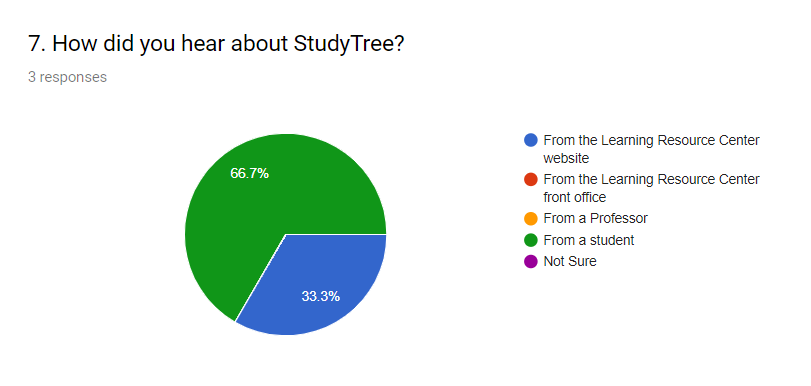


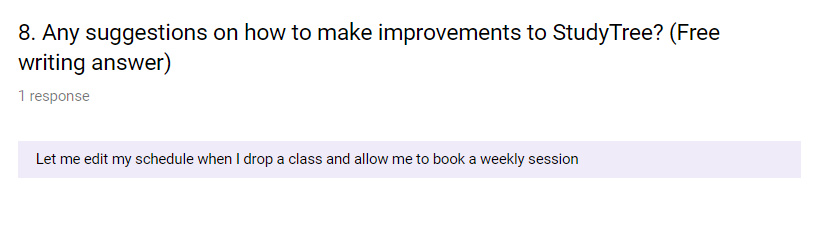












**Requirements**

* Functional

**Process-Oriented:**

1. Schedule Updating

1.1 The app would let the tutors go in and change their schedule around

1.2 Tutors can see how many students they are allowed to tutor weekly.

1.3 The app will allow tutors choose which hours they want to be “Available for booking” and which hours to be “Potential hours”

1. Appointments

2.1 The app will let the student the number of allowed appointment per week and per semester.

2.2 The app will send appt. Confirmation and reminders which can be text or email

**Information-Oriented:**

1. Registration Information

1.1 The app will have a AU email registration only.

1.2 The app will be able to create tutor/student profiles

1.3 The app will have a waiting list for booked sessions

1. Forms

2.1 The app will have forms that will be easily customized (reports,  
 appts.)

2.2 The app will allow students to search topics or subjects.

1. Reports

3.1 The app will have reports for daily, weekly, monthly and yearly activity.

* Non-functional

**Operational:**

1.1 The app will allow different start/end times and length of appointments

1.2 The app will generate reports and analytics

1.3 The app will be accessible to students on multiple devices and locations when wanting to change schedule or just schedule

**Performance:**

1.1 The app will have great customer support

1.2 The app will update every 15 minutes

1.3 The app will be available 24 hours a day

**Security:**

1.1 The app will track no-shows and block offenders after 2 consecutive No-shows

**Cultural and political:**  1.1 The app will have a survey option for students to take for feedback at the end of every semester.

**Use Case Analysis**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case Name: Registration** | | | **ID: 1** | | **Priority: High** | |
| **Brief Description:** This use case will have umbc email registration only, and will help create the tutor, student and supervisors profiles. It will also, able the students to be able to be waitlisted for booked sessions so they don’t have to keep checking the app for booked session to open up. Supervisors will be able to keep tract/ interact with students and tutors. | | | | | | |
| **Actor:** Students, Tutors and Supervisors | | | | | | |
| **Trigger:** Students can use the app to book a session with a Tutor and a Tutor can register in order to be matched up with a student. | | | | | | |
| **Type:** External | | | | | | |
| **Preconditions:**   * Student, tutor and supervisor must have a functioning umbc email * Tutors must be hired by the UMBC supervisor * Students, tutors, and supervisors must remember their login information * Supervisors must be authorized by UMBC to monitor tutors and students | | | | | | |
| **Normal Course**   1. Students and Tutors register with their UMBC email 2. A registration confirmation is sent to their email 3. Students/tutors can login with their email and password to on the app 4. Students can be waiting list for booked sessions 5. Supervisor can assign tutoring classrooms to tutors | | | | **Information for Steps**  → register for the app  ← confirmation information is sent  → registered users can logback in with their username/password  →students can be waitlisted for registration  → students books a session with a tutor, then classrooms for tutoring are assigned by supervisor to tutors | | |
| **Postconditions:**   * When waitlisted student is able to register for a session a reminder is sent via email. * Students should remember they login information for future login purposes * A classroom where a student and a tutor can meet is set | | | | | | |
| **Summary:**  **Inputs Source Outputs Destination** | | | | | | |
| Student and Tutor registration information | Student and Tutor | Registration is complete and | | | | student , tutor, supervisor |

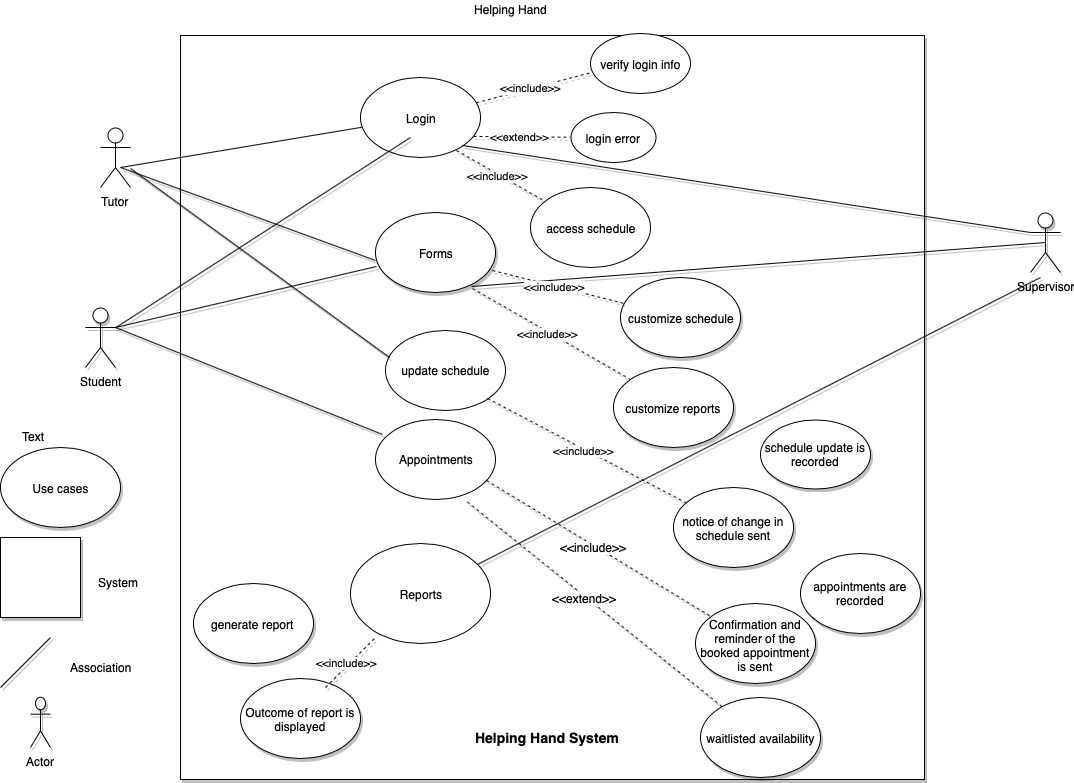
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case Name: Forms** | | | **ID: 2** | | **Priority: High** | |
| **Brief Description:** This use case describes how the app will have forms that can be customized to the students/tutors/supervisors needs. This can be used by students and tutors to customize their own appointments as well as the subjects they are interested in. The supervisor will also, be able to customize reports | | | | | | |
| **Actor:** students, tutors and supervisors | | | | | | |
| **Trigger:** The need of customizing of personal preference in regards tp appt and courses and reports | | | | | | |
| **Type:** External | | | | | | |
| **Preconditions:**   * Students, tutors, and supervisors must have be registered users of the app * There must have students and tutors that are actively engaged in order for a supervisor to be able to customize reports | | | | | | |
| **Normal Course**   1. Students and tutors can customize their schedules and courses before booking any sessions 2. Supervisors can customize reports before generating them | | | | **Information for Steps**  → students/tutors must login to the app  → customize schedule within the app  ← students and tutors are able to see their customized/updated schedule and courses  → supervisors must login to the app  →they can customize reports before generating them  ←customized reports are visible supervisors | | |
| **Postconditions:**   * Report has been generated | | | | | | |
| **Summary:**  **Inputs Source Outputs Destination** | | | | | | |
| Students/tutors schedules and preferred courses  Supervisors need for customizing reports | Student  Tuttor  Supervisor | Customized schedule and courses for students and tutors  Customized report | | | | Student  Tutor  Supervisor |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case Name: Updating Schedules** | | | **ID: 3** | | **Priority: High** | |
| **Brief Description:** This use case describes how the app will allow autonomy for tutors in regards to changing their schedule, the amount of hours they would like, and the number of students they will see per week. | | | | | | |
| **Actor: Tutors** | | | | | | |
| **Trigger:** Tutors preparation for their work week. This incorporates changes in schedule, and in hours to work. | | | | | | |
| **Type:** External | | | | | | |
| **Preconditions:**   * The tutor need to login into the system using a umbc id and password * The tutor must have an emergency change in schedule. * The tutor must know how many hours they would like to register for. | | | | | | |
| **Normal Course**   1. Tutors are able to access their schedule 2. Tutors can change their schedule    1. If they would like more hours, it can be checked in the database.    2. If they would like to release some tutoring session, student will receive a notice of a schedule change. 3. Changes in schedule will be recorded in the system. | | | | **Information for Steps**  ← log into system  → access to database  → available hours displayed  → notice of change in schedule sent  ← change in schedule for tutor  → new record processed in database | | |
| **Postconditions:**   * Tutor has a revised schedule * Notice is sent to student of revision if it pertains to the student. | | | | | | |
| **Summary:**  **Inputs Source Outputs Destination** | | | | | | |
| Tutor Access | Tutor | Revised schedule with notices being sent to students regarding the change. | | | | Student |

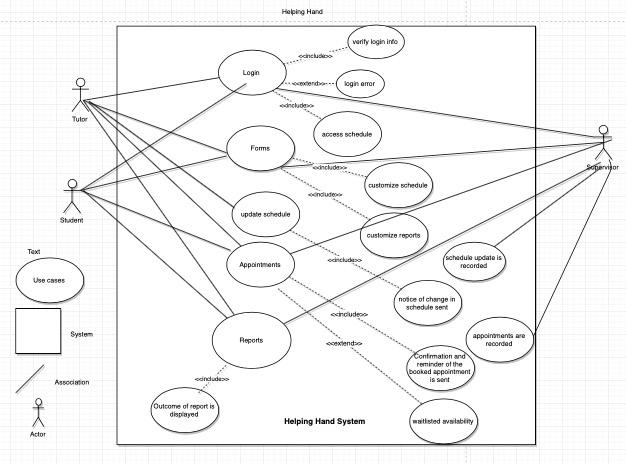
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case Name: Appointments** | | | **ID: 4** | | **Priority: High** | |
| **Brief Description:** This use case will describe how and how many tutoring appointments can the students make per week with its confirmation as well as reminder via text or email. | | | | | | |
| **Actor:** Students | | | | | | |
| **Trigger:** Helping students book an appointment on time and remind them about the appointments. | | | | | | |
| **Type:** External | | | | | | |
| **Preconditions:**   * Student will need to login with their student id and password. * The students should know the limitation of appointment. * Student should select the means of reminder i.e. text or email. | | | | | | |
| **Normal Course**   1. Students can book the available appointments. 2. The appointment is received and acknowledged. 3. Appointments will be recorded in the system. 4. Confirmation and reminder of the booked appointment is sent. 5. Students can be waitlisted for tutoring sessions | | | | **Information for Steps**  ← log into the system  → access to the database  ← available/allowed appointment slot displayed  ←Needed appointment booked  →Confirmation of reserved appointment sent  →revised appointment slot displayed  →Reminder of an appointment sent  ← reminder of available/open previously waitlisted session will be sent | | |
| **Postconditions:**   * Students know the limitation of appointment and schedule on time and only when needed. * Confirmation is sent after booking an appointment. * Reminder is sent via text/email before the appointment time. | | | | | | |
| **Summary:**  **Inputs Source Outputs Destination** | | | | | | |
| Student information | Student | Necessary appointments made and a confirmation and reminder sent to the students. | | | | Tutor |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Use Case Name: Reports** | | | **ID: 5** | | **Priority: High** | |
| **Brief Description:** This use case describes how the app will have generate reports for the supervisor to analyze | | | | | | |
| **Actor:** Supervisor | | | | | | |
| **Trigger:** It is the end of a day, week, month and year | | | | | | |
| **Type:** Temporal | | | | | | |
| **Preconditions:**   * App can track activities done by students and tutors * Supervisor can customize reports | | | | | | |
| **Normal Course**   1. Daily, weekly, monthly and yearly reports are generated | | | | **Information for Steps**  → Supervisor requests customized report  ←Outcome of reports is displayed to supervisor | | |
| **Postconditions:**   * Reports have been generated based on the supervisors need | | | | | | |
| **Summary:**  **Inputs Source Outputs Destination** | | | | | | |
| Request of reports by supervisor | Supervisor  App Database | Reports are given by the app based on daily, weekly, monthly, and yearly basis | | | | Supervisor |

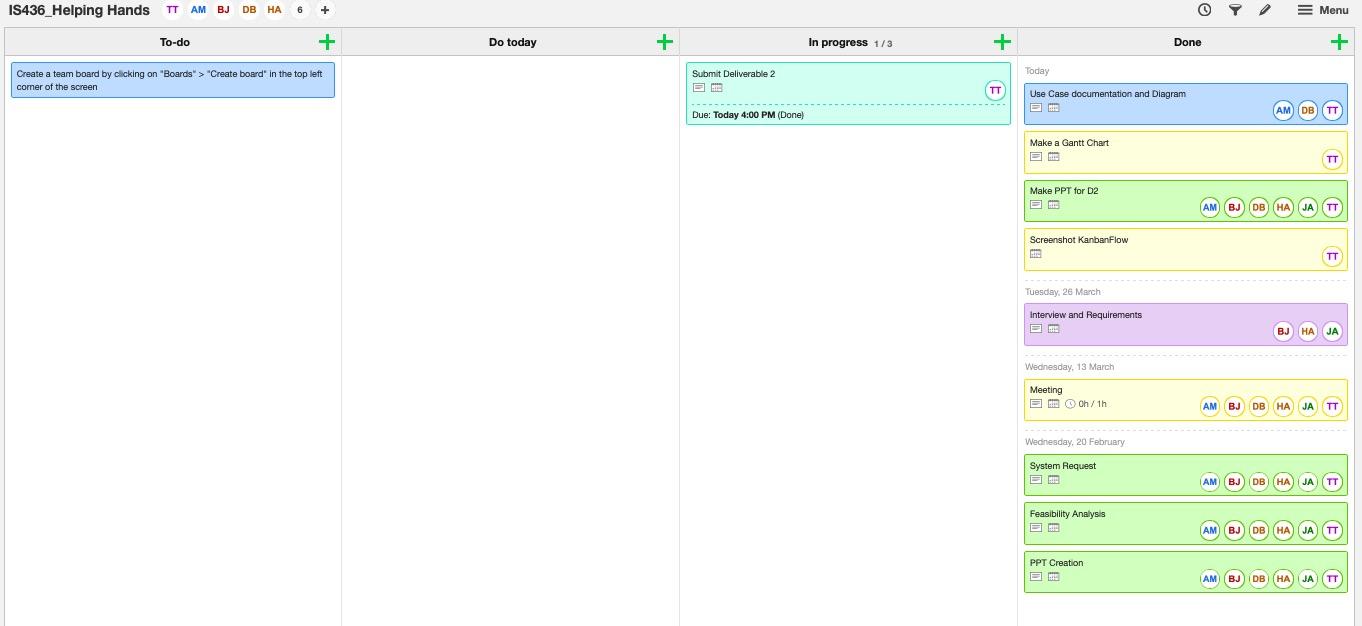
USE CASE DIAGRAM:



Revised Use Case Diagram:

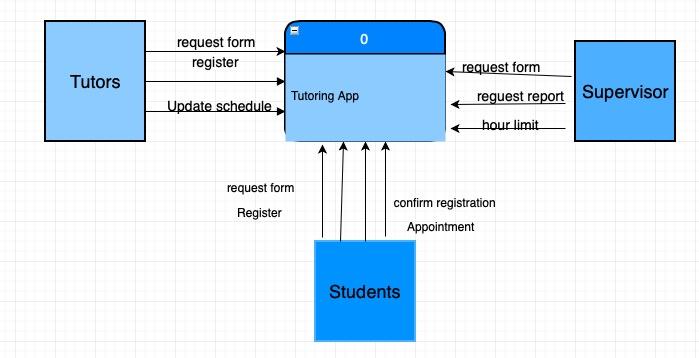


Kanbanflow



**Deliverable 3:Process Modeling**

Context DFD



Text Definitions: Context DFD

**Process:**

* **“Tutoring App”** It is the system that carries everything inside of it.

**Entity:**

* **1st Entity** = “Tutor”: Is the person that provides help to the students in the area that they need help in.
* **2nd Entity** = “Students”: They are the customer that goes to the tutors when needing help in a certain subject or area.
* **3rd Entity**  = “Supervisor”: Is the person that will oversee the whole business of what is going on between the student and tutors.

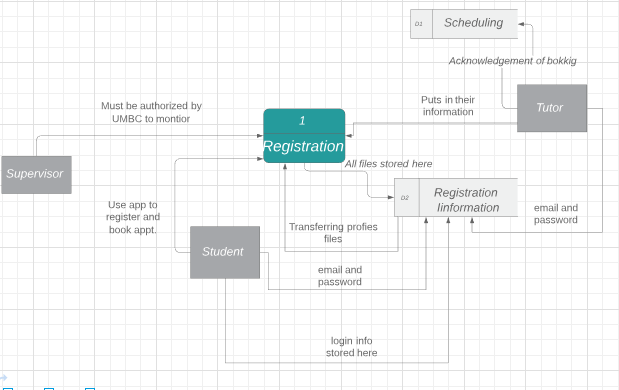
**DataStore:**

* **1st DataStore** = “Tutoring App” stores all the data of the app including tutors and students profiles and schedules.

**Data Flow: (left to right)**

* **1st Data flow**: Coming from tutor entity to the process “Tutoring App”= “Request Form.” Tutors have to be registered to be able to request forms.
* **2nd Data flow**: Coming from tutor entity to the process “Tutoring App”= “register”: Tutors have to have umbc email address to be able to register.
* **3rd Data flow**: Coming from tutor entity to the process “Tutoring App”= “update schedule” Tutors have to be registered to be able to edit their schedule.
* **4th Data flow**: Coming from student entity to the process “Tutoring App”= “register”: students have to have umbc email address to be able to register.
* **5th Data flow**: Coming from student entity to the process “Tutoring App”= “request form”: students have to be registered to be able to request forms.
* **6th Data flow**: Coming from student entity to the process “Tutoring App”= “appointment”: students have to be registered to be able to request an appointment.
* **7th Data flow**: Coming from student entity to the process “Tutoring App”= “confirm registration”: students have to confirm their registration through the email sent to their email address to be able to continue.
* **8th Data flow**: Coming from supervisor entity to the process “Tutoring App”= “request report”: supervisors have to be registered and they must have hired at least one tutor to be able to request a report.
* **9th Data flow:** Coming from supervisor entity to the process “Tutoring App”= “request form”: supervisors have to be registered to be able to request forms.
* **10th Data flow:** Coming from supervisor entity to the process “Tutoring App”= “hour limit”: supervisors have to have at least one tutor to be able to put restrictions on the number of hours they are allowed to tutor weekly.

Level 0 DFD - Registration



Text Definitions: Registration

**Process:**

* **“Registration”** It is what we are working towards in order to better the old system. The system carries everything inside of it.

**Entity:**

* **1st Entity** = “Tutor”: Is the person that provides help to the students in the area that they need help in.
* **2nd Entity** = “Students”: They are the customer that goes to the tutors when needing help in a certain subject or area.
* **3rd Entity**  = “Supervisor”: Is the person that will oversee the whole business of what is going on between the student and tutors.

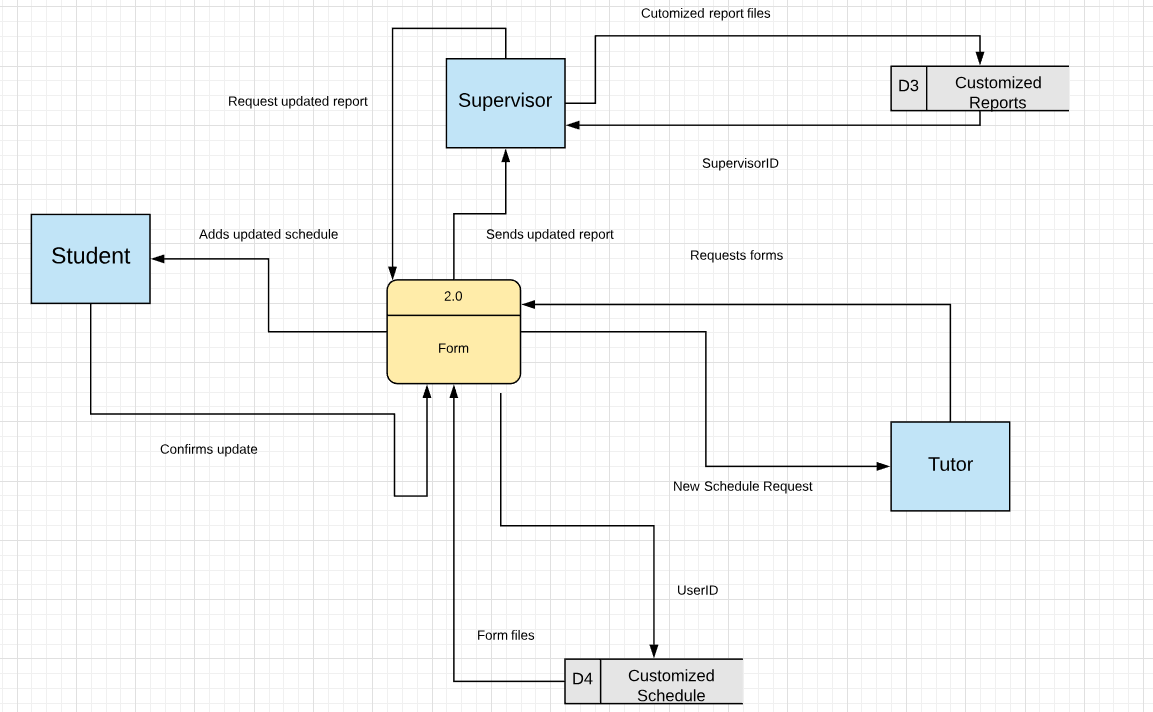
**DataStore:**

* **1st DataStore** = “D1 - Scheduling” stores all of the appointments that are formed by the tutors and students.
* **2nd Data Store** = “D2 - Registration Information” stores all of the profiles and personal information of the student and tutor.

**Data Flow: (left to right)**

* **1st Data flow**: coming from supervisor entity to process 1 = “Must be authorized by UMBC to monitor.” has to register so that they can oversee and keep track of what is going on.
* **2nd Data flow**: coming from process 1 to student entity = “Use app to register and book appt.” after the registration process the student can book an appt.
* **3rd Data flow**: coming from profiles to process 1 = “Transferring profile files” once the profiles from the tutor and student is created it is then brought through the registration process in order to process in the app.
* **4th Data flow**: coming from process 1 to registration information data store = “All files stored here” contains all of the processed registered profiles.
* **5th Data flow**: coming from tutor entity to process 1 = “puts in their information” allows them to register their information on the app.
* **6th Data flow**: coming from tutor entity to scheduling data store = “Acknowledgement of booking” allows the tutor appts scheduled to be stored in the app database.
* **7th Data flow**: coming from tutor entity to profiles= “email and password login” allows the tutor and student login info to be processed.
* **8th Data flow**: coming from profiles to registration information data store = “login info store here”stores all of the login requirements and info that is given and sent.
* **9th Data flow:** coming from student entity to D2 registration information = “ email and password” allows the tutor and student login info to be processed.

Level 0 DFD - Forms (Use case 2)



**Text Definition(Level 0 DFD) Reports**

**Entity:**

1st Entity = “Tutor”: Is the person that provides help to the students in the area that they need help in.

2nd Entity = “Students”: They are the customer that goes to the tutors when needing help in a certain subject or area.

3rd Entity = “Supervisor”: Is the person that will oversee the whole business of what is going on between the student and tutors.

**Datastore:**

D3 = "Customized reports" is a data store that will hold report files that the supervisor wanted to hold. Only the supervisor will have permission for that data.

D4= "Customized Schedule" is a datastore that will hold files from tutors and students. They will keep the ideal appointments and the way to access is through their login Id and password.

**Data Flow: (starting from left to the right)**

1st data flow: "Adds updated schedule" is when the system form returns the updated schedule to the student.

2nd data flow: "Confirm update" the system will reply back to the student request by outputting available days and times for the courses they wanted to be tutored.

3rd data flow: "Requests updated report" is when the supervisor wants to get the data from the system about the daily appointments

4th data flow: "sends updated report" is when the system responds from the supervisor request to get a daily or weekly report on appointments.

5th data flow: "Form files" the form will give the files to the data store customized schedule.

6th data flow: "UserId" will give permission to access their own file when they enter the correct login and password from the app.

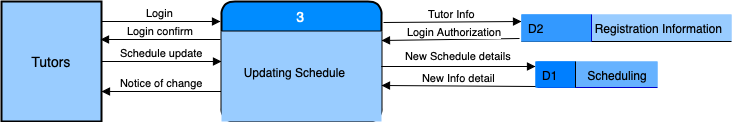
7th data flow: "Request forms" is when the tutor wants to update the available dates he or she is able to teach.

8th data flow: " New Schedule " is when the tutor is given a response from the system on when to teach and which subject.

9th data flow: "customized report files" is when the supervisor will want to insert their customized report files to the data store.

10th data flow: "SupervisorID" is when the datastore will give an unique login info the supervisor. Even though the supervisor is the only one using the datastore, giving a login and password is ideal to protect information.

**Level 0 DFD: Updating Schedule (Use Case3)**



**Text Definitions(Level 0 DFD: Updating Schedule**

**Process:**

* **1st process** =“Updating Schedule” this allows the tutor to update their schedule if any change occurs to their availability throughout the semester.

**Entity:**

* **1st Entity** = “Tutor”: a person who works as a guide to student to better their academic achievement

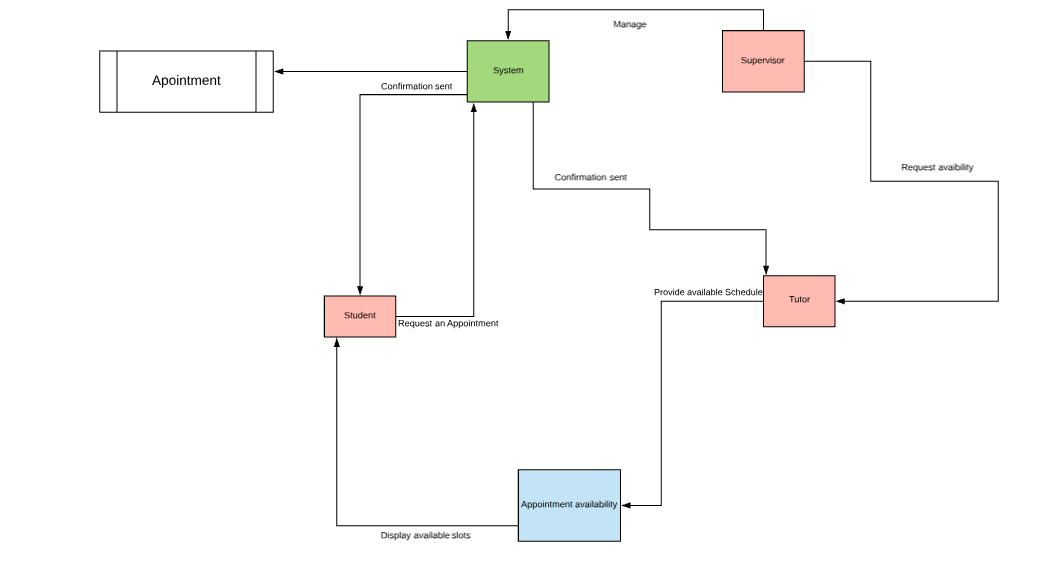
**DataStore:**

* **1st DataStore** = “D1 - Scheduling” stores the updated schedule given by tutors and replaces their old schedule
* **2nd Data Store** = “D2 - Registration Information” this is where the tutors information is stored

**Data Flow:(**left to right)

* **1st Data flow**: coming from tutor entity to process 3 = “Login” the tutor have to login in order to update their schedule
* **2nd Data flow**: coming from process 3 to D2= “Tutor Info” after the tutor attempts to login, the login information is checked again the database information
* **3rd Data flow**: coming from D2 process 3= “Login Authorization” this authorized the login of the turot if the info matches what is stored in the D2 database.
* **4th Data flow**: coming from process 3 to tutor entity is = “Login Confirm” this lets the user to receive a message that says they are able to login to the system
* **5th Data flow**: coming tutor entity to process 3 is = “Scheduling update” tutor submits hew/his update schedule information
* **6th Data flow**: coming from process 3 to D1:scheduling is = “New Schedule Details” this allows the database to store the update schedule and replace old scheduling information
* **7th Data flow**: coming from D1:scheduling to process 3 = “New Info Detail” this sends a message of the updated schedule in the system
* **8th Data flow**: coming from process 3 to tutor entity is = “Notice of Change” this is a message displayed to the tutor about whether their schedule has been updated or not.

**Level 0 DFD: Appointment(Use Case 4)**

****

**Text Definitions(Level 0 DFD): Appointment**

**Process:**

- **1st process =** We are working on“Appointment” in order to make a easy and flexible appointment system. The system carries everything inside of it.

- **2nd process =** “Appointment availability”: is where students can check for next available appointments and the requested appointment reports are sent to the system process.

- **3rd process =** “Students”: will be able to check the appointment availability and book a new appointment as needed.

- **4th process =** “Tutor”: will be able to provide their own schedule as availability for tutoring and will receive the confirmation every time when appointed.

- **5th process =** “Supervisor”: will manage and handle the entire system.

**Entity:**

- **1st Entity =** “Supervisor”: Is the person who will manage the overall system and changes in the appointment.

- **2nd Entity =** “Tutor”: Is the person who will provide their schedule for the appointment and help student.

- **3rd Entity =** “Students”: They are the customers who will book an appointment with the tutors as needed.

**Data Store:**

- **Datastore1 =** “Appointment” stores and generates all the requested and appointed appointments by the students and tutors.

**Data Flow:**

**- 1st Data Flow:** 1st data flow going into “Tutor” from “Supervisor” is for supervisor to request the availability of all the tutors to set up an appointment schedule.

- **2nd Data Flow:** 2nd data flow going into “Appointment availability” from “Tutor” this is when tutors provide their available hours to set up an appointment schedule.

- **3rd Data Flow:** 3rd data flow is going from “Appointment availability” to the “”System” here the report of tutors availability is sent to the system set up an appointment schedule for students.

- **4th Data Flow**: 4th data flow is going into “Students” from “Appointment availability” this is when the students will be displayed the available schedule.

- **5th Data Flow:** 5th data flow is going into “System” from the “Student” this is when student request or select their appointment with the tutors as needed or as available.

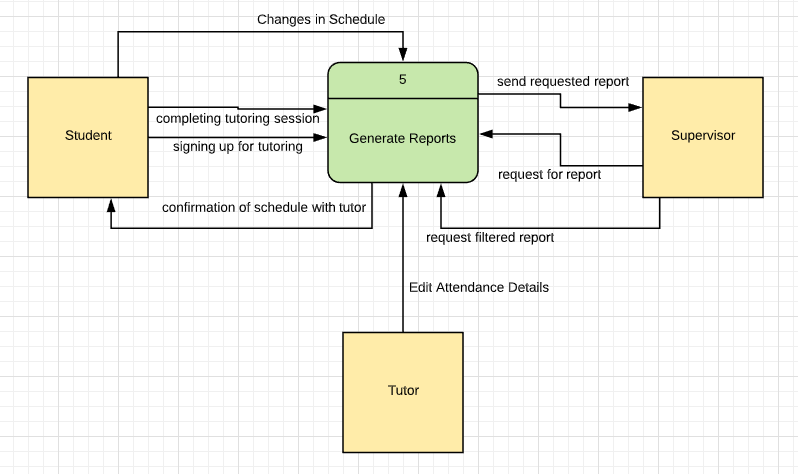
- **6th Data Flow:** 6th data flow is going into “Tutor” from “System” this is when the System sends a notification/confirmation to the tutor via text/email for the tutoring appointment.

- **7th Data Flow:** 7th data flow is going into “Students” from “System” this is when the System approves the students request for the appointment and sends the confirmation via text/email.

- **8th Data Flow:** 8th data flow is going into “Appointment” data store from the “System” when all the process are finished and finally the appointment is created and updated.

- **9th Data Flow:** 9th data flow is going into “System” from “Supervisor” this is when any changes are made after the “Appointment” is already created, where the Supervisor will handle and manage the changes.

**Level 0 DFD - Reports (Use Case 5)**



**Text Definitions(Level 0 DFD): Reports**

**Process**:

**Reports**: It creates reports for all tutoring sessions that are created, completed, and canceled.

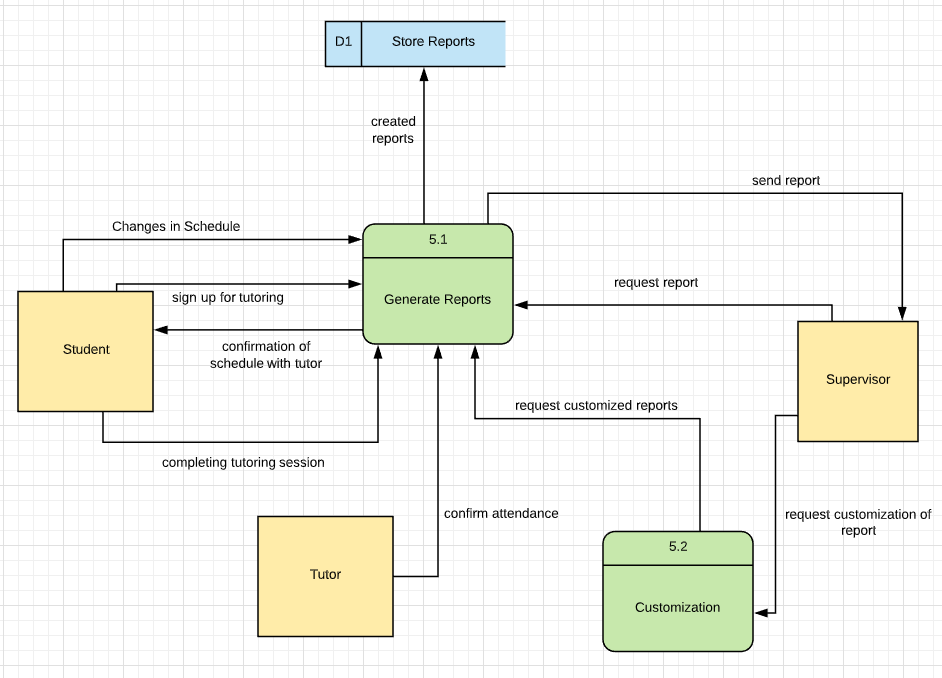
**Entity**:

* **1st Entity** =“Students”: They are the customer that goes to the tutors when needing help in a certain subject or area.
* **2nd Entity** = “Supervisor”: Is the person that will oversee the whole business of what is going on between the student and tutors.
* **3rd Entity** = “Tutor” : is the person that will be conducting the tutoring session as well as make updates to reports.

**Data Flow:**

* **1st data flow:** going into Generate Reports(process) from the student is = “signing up for tutoring.” This is where it starts to create the report for any tutoring session that is requested.
* **2nd data flow:** going into the Generate Reports(process) from the student is = “ completing tutoring session.” This step is used to complete the reports in all the tutoring sessions that have occurred. Essentially, this is based on whether or not the session had taken place and if it was completed.
* **3rd data flow**: going into the Generate Reports(process) from the student is = “changes in schedule.” This step is to record any changes the students make after signing up for the tutoring session.
* **4th data flow**: going into the Student entity from the Generate Reports(process) is =”confirmation of schedule with tutor.” This step acts as a trigger based on the schedule that was generated from the student and tutor. It will ask for a confirmation from the student so the report can be held accountable in its acts of solidifying the tutoring session.
* **5th data flow**: going into the Generate Reports (process) from the supervisor is = “request for report.” This is an action that is taken place by the supervisor so that they can see an outline of the report that is generated for any particular tutoring session.
* **6th data flow:** going into the Generate Reports (process) from the supervisor is = “request filtered report.” This action is only used if the supervisor wanted to filter specific criteria in some or all reports such as, those who have not completed their tutoring session. These reports will have include the reports of those who have not completed their tutoring session.
* **7th data flow:** going into the supervisor entity from the Generate Reports(process) is = “send requested report.” This action is sent to the supervisor based on the action the supervisor had taken in viewing reports.
* **8th data flow:** going into the Generate Reports from the tutor entity is = “edit attendance details.” The tutor will be able to mark up the student if the student did or did not attend the tutoring ses

**Level 1 DFD - Reports (Case Study 5)**



Text Definitions(Level 1 DFD): Reports

**Process**:

**Reports**: It creates reports for all tutoring sessions that are created, completed, and canceled.

**Customization:** This process goes a bit further in the process of creating reports. This process allows the supervisors to create specific reports according to their own criteria.

**Entity**:

* **1st Entity** =“Students”: They are the customer that goes to the tutors when needing help in a certain subject or area.
* **2nd Entity** = “Supervisor”: Is the person that will oversee the whole business of what is going on between the student and tutors.
* **3rd Entity** = “Tutor” : is the person that will be conducting the tutoring session as well as make updates to reports.

**Data Store:**

* **1st Entity** = “Store Report”: This data store is used to keep and retain all reports that are generated by the app.

**Data Flow:**

* **1st data flow:** going into Generate Reports(process) from the student is = “signing up for tutoring.” This is where it starts to create the report for any tutoring session that is requested.
* **2nd data flow:** going into the Generate Reports(process) from the student is = “ completing tutoring session.” This step is used to complete the reports in all the tutoring sessions that have occurred. Essentially, this is based on whether or not the session had taken place and if it was completed.
* **3rd data flow**: going into the Generate Reports(process) from the student is = “changes in schedule.” This step is to record any changes the students make after signing up for the tutoring session.
* **4th data flow**: going into the Student entity from the Generate Reports(process) is =”confirmation of schedule with tutor.” This step acts as a trigger based on the schedule that was generated from the student and tutor. It will ask for a confirmation from the student so the report can be held accountable in its acts of solidifying the tutoring session.
* **5th data flow**: going into the Generate Reports (process) from the supervisor is = “request for report.” This is an action that is taken place by the supervisor so that they can see an outline of the report that is generated for any particular tutoring session.
* **6th data flow:** going into the Customization (process) from the supervisor is = “request customization of report.” This action is only used if the supervisor wanted to filter specific criteria in some or all reports such as, those who have not completed their tutoring session.
* **7th data flow:** going into the supervisor entity from the Generate Reports(process) is = “send report.” This action is sent to the supervisor based on the action the supervisor had taken in viewing reports. This is the standard report that is given to the supervisor based on one session.
* **8th data flow:** going into the Generate Reports(process) from the Customization(process) is = “request customized reports.” This causes the generate report process to actually carry out the filters and specific criteria the supervisor requested.
* **9th data flow:** going into the store reports(data store) from the Generate Reports(process) is = “created reports.” This data store will store all of the reports that are generated from the Generate Reports process.
* **10th data flow:** going into the Generate Reports from the tutor entity is = “edit attendance details.” The tutor will be able to mark up the student if the student did or did not attend the tutoring session.

**Deliverable 4 – “Data Modeling and Starting Design”**

# **ERD:**

# 

# 

# 

# 

# **ERD Descriptions:**

**Entities:**

* **1nd Entity** = “Tutor”: is the record of the tutor
  + Includes tutor\_name, tutor\_email, tutor\_courses, tutor\_availability and tutoring\_id (primary key)
* **2rd Entity** = “Tutoring\_Room”: this records the room information of where the tutoring is going to happen
  + It includes room\_location, room\_number, room\_capacity and room\_id (primary key).
* **3th Entity** = “Supervisor”: records about the information of the supervisor of the tutoring center.
  + It includes a primary key of supervisor\_id, and attributes such as supervisor\_name, supervisor\_emial, asupervisor\_department.
  + It has also a foreign key from the Tutoring\_Room entity called Room\_ID
* **4th Entity** = “Student”: records information about student
  + Includes student\_name, student\_email, student\_course and a primary key called student\_id
* **5st Entity** = “Session”: records about a session that is booked.
  + It has four primary keys session\_id, student\_id, tutor\_id, and room\_id
  + student\_id, tutor\_id, and room\_id also acts as a foreign key from the tutor, studnet, and tutoring\_room tables.

**Relationships:**

* The events involves only one account. Whereas, the account incorporates that there can be a zero-to- many events accessible to it.
* Tutor: can have 0 or more sessions booked, while a session must have a tutor booked for it.
* Tutor: can have 1 or more supervisors, while a supervisor can have one or more tutors he/she manages
* Supervisor: can book 0 or more tutoring rooms, whereas a tutoring room can be booked by one or more supervisors
* Supervisor: can manage 0 or more students, whereas a student can have one or more supervisors.

-Student: can book a session 0 or multiple times, while a scissor must have one

have one student to be booked.

-Session: must have a student, a tutor, and a designated tutoring\_room to be able to booked.

-Tutoring\_room can have 0 or multiple booked sessions.

**Briana’s Alternative Matrix:**

# 

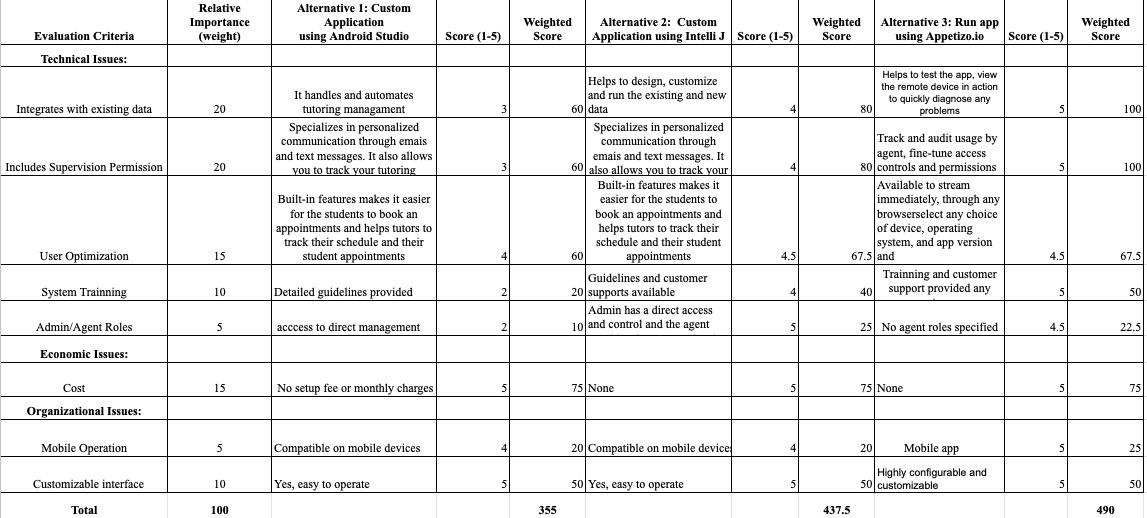
**Final Result:**

Alternative 1: 390 points

Alternative 2: 370 points

**Alternative 3: 550 points**

**Timnit’s Alternative Matrix:**

****

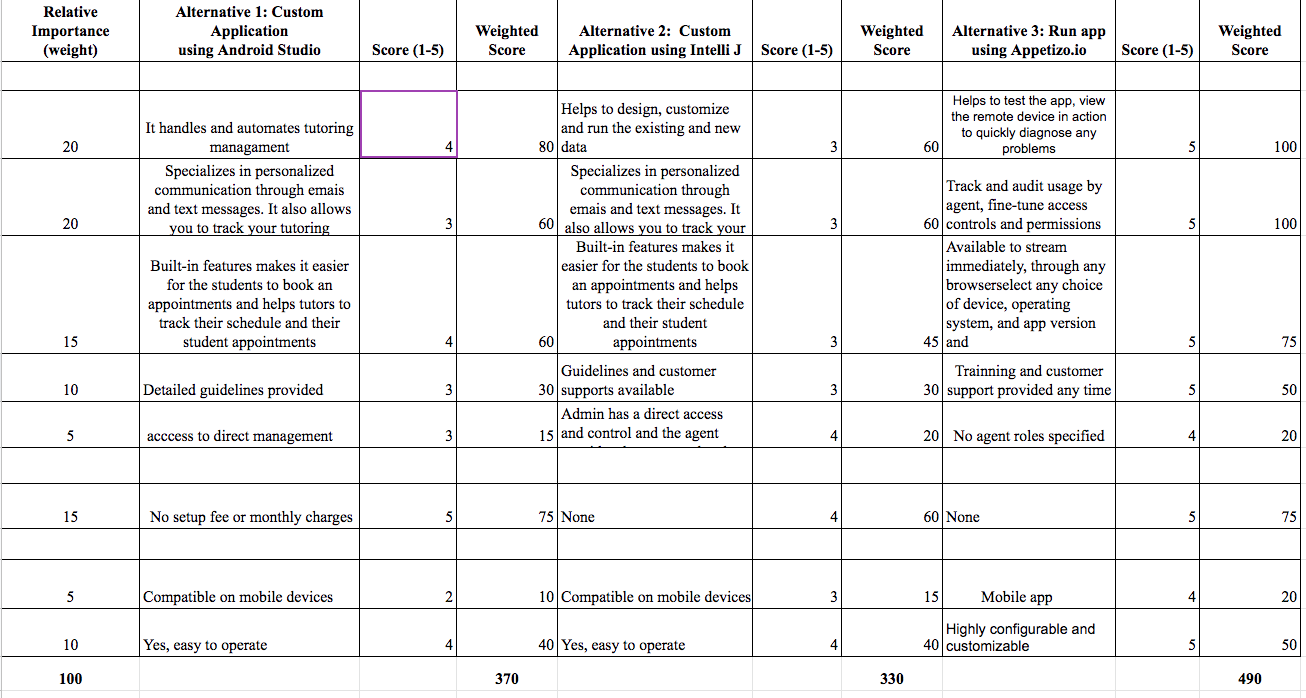
**Final result:**

Alternative 1: 355 points

Alternative 2: 437.5 points

**Alternative 3: 490 points**

**Anusha’s Alternative Matrix:**

****

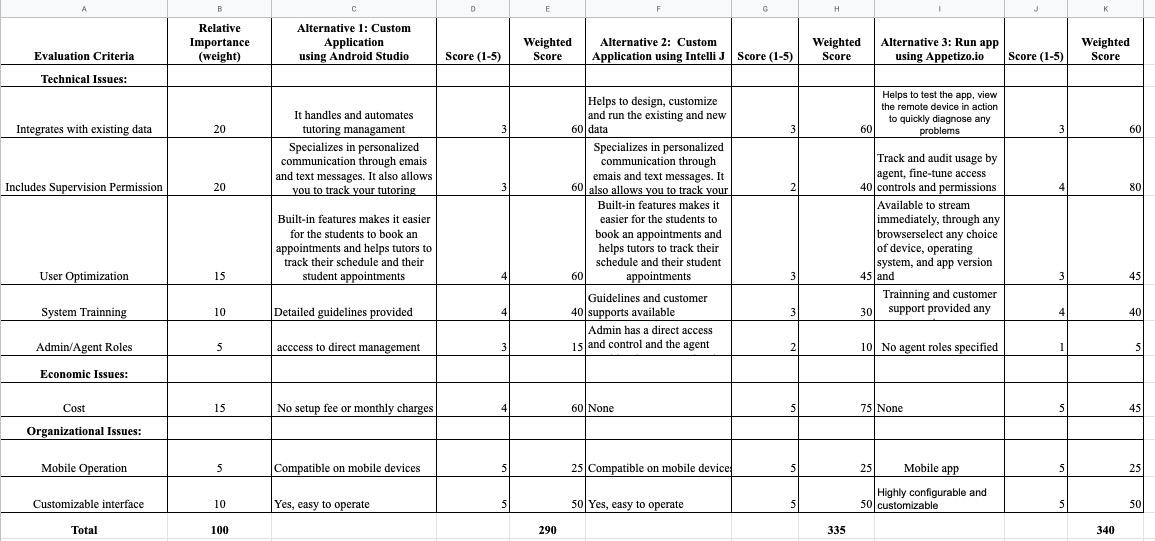
**Final result:**

Alternative 1: 370 points

Alternative 2: 330 points

**Alternative 3: 490 points**

**Haythem’s Alternative Matrix:**



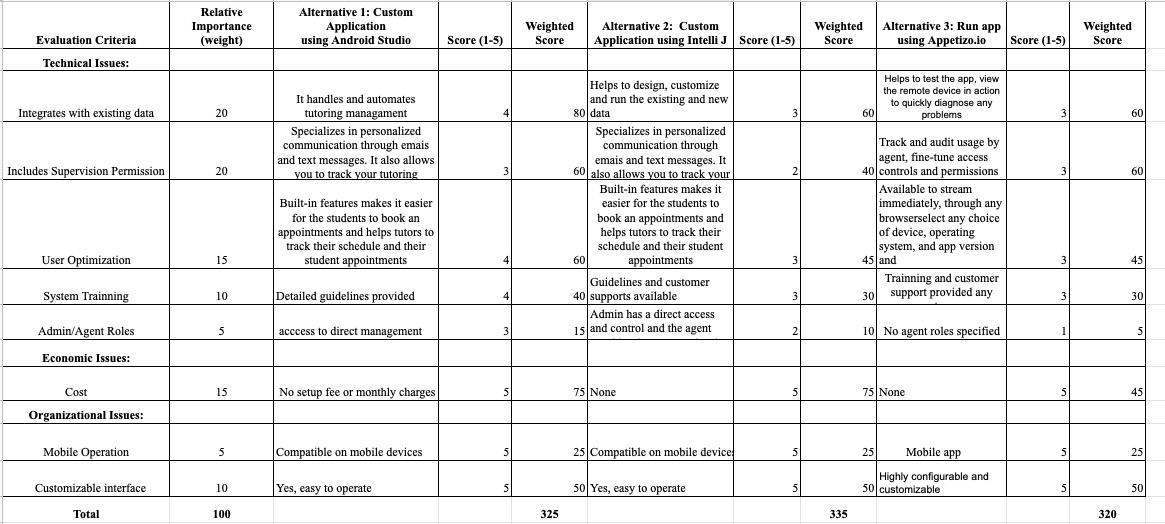
**Final result:**

Alternative 1: 290 points

Alternative 2: 335 points

**Alternative 3: 340 points**

**Daniel’s Alternative Matrix:**



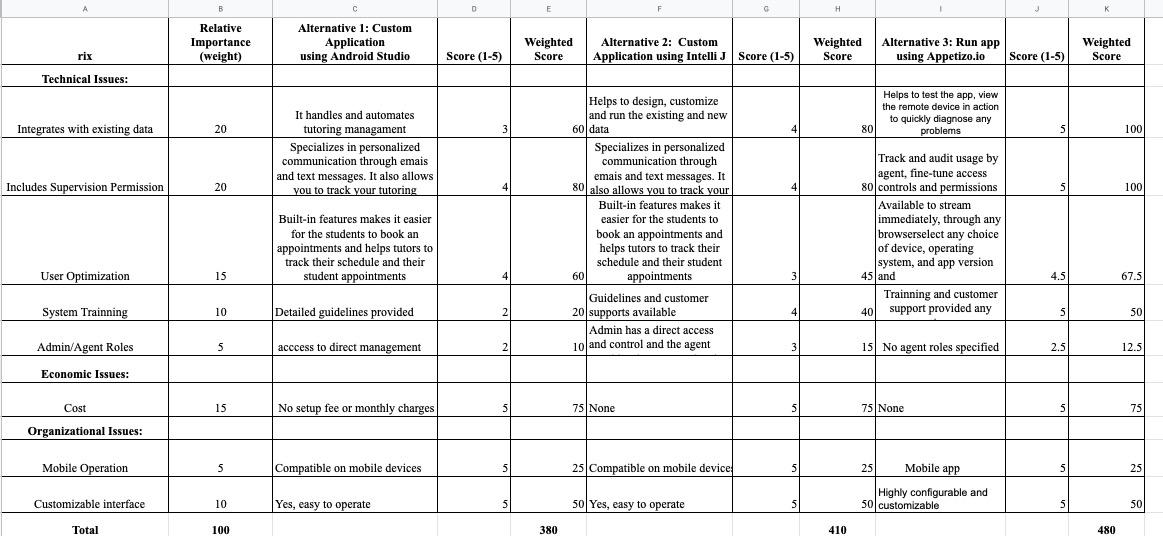
**Final result:**

Alternative 1: 325 points

Alternative 2: 335 points

Alternative 3: 320 points

**James Alternative Matrix:**



**Final Result:**

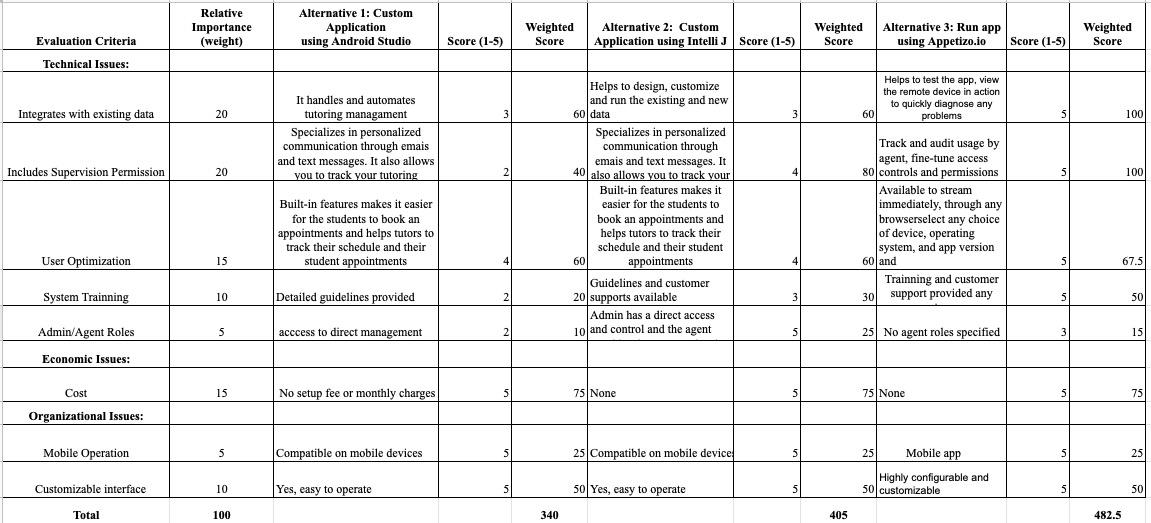
Alternative 1: 380 points

Alternative 2: 410 points

**Alternative 3: 480 points**

**Team’ Alternative Matrix:**

*(For more clarity of pic look at attachment: Alternative Matrix \_D4)*



**Final Result:**

Alternative 1: 340 points

Alternative 2: 405 points

**Alternative 3: 482.5 points**

# 

# **Team Matrix Description:**

After reviewing all of the team choices based off of the original team matrix, our team has landed on the highest ranking alternative. Almost every group member voted on alternative 3 being appetizo.io . The biggest reasons why alternative 3 got the higher points were because of system training and tracks the usage. The application will give system training to the tutors,supervisors and the students. It’ll also give more trackage to the users so the supervisors can generate detailed reports. Each member agreed on which alternative was the best of the three which ended up being the first alternative. Running out Helping Hands’ app through the Appetizio.io software meets all our teams need. We came to this conclusion based on the needs of our app and our potential future customers. Appetizio.io can diagnose any problems our app encounters. Also, it is highly configurable and customizable making it very easy for first timers to familiarize themselves in a short amount of time. It can be also used remotely to diagnose any problems that are seen in out app. Most importantly, it can be used by any device based on the customers choice, operating system making it very efficient and it doesn't cost money. Thus, making it more affordable.

**Architecture Matrix:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Requirements** | **Server-Based** | **Client-Based** | **Thin Client-Server** | **Thick Client-Server** |
| **Operational Requirements** |  |  |  |  |
| System Integration Requirements |  |  | X |  |
| Portability Requirements | X |  |  |  |
| **Performance Requirements** |  |  |  |  |
| Speed Requirements |  |  | X |  |
| Availability/Reliability Requirements | X |  | X | X |
| **Security Requirements** |  |  |  |  |
| Access Control Requirements | X |  |  |  |
| **Cultural/Political Requirements** |  |  |  |  |
| Explicit Unstated Norms |  |  | X |  |
| Legal Requirements | X |  |  |  |

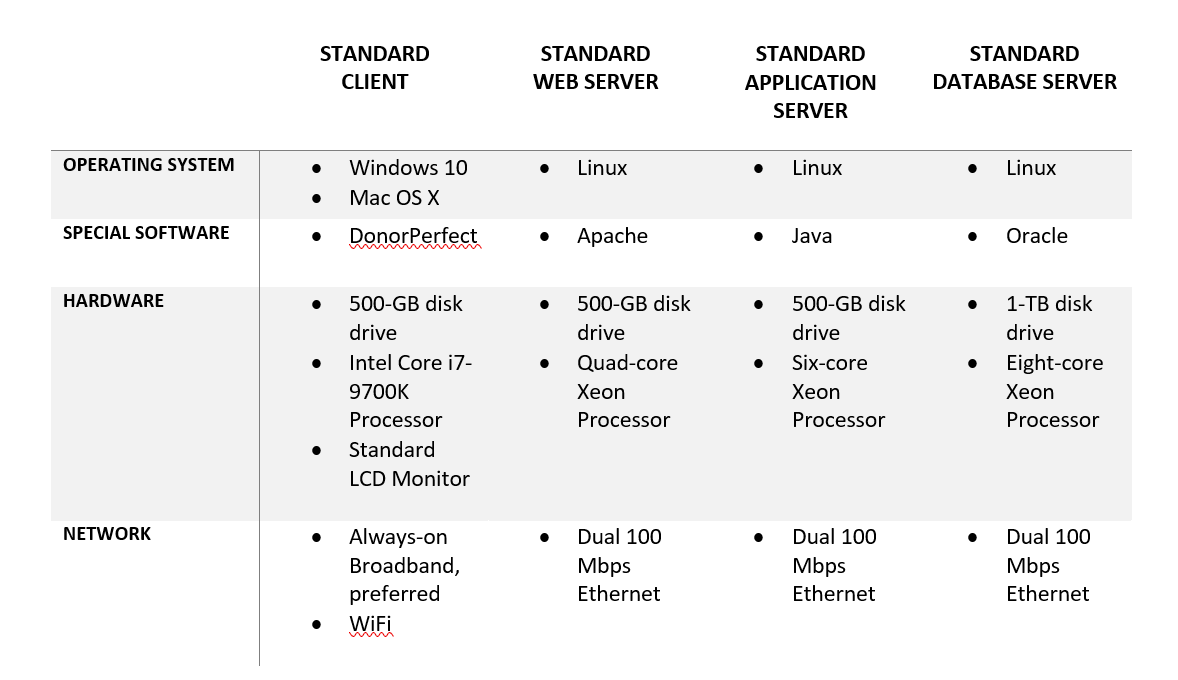
# **Architecture Matrix Description**

Based on our architecture matrix, we have determined which architecture will be the best fit. In this particular case, server-based architecture is a good fit for the type of internal system we are trying to build.

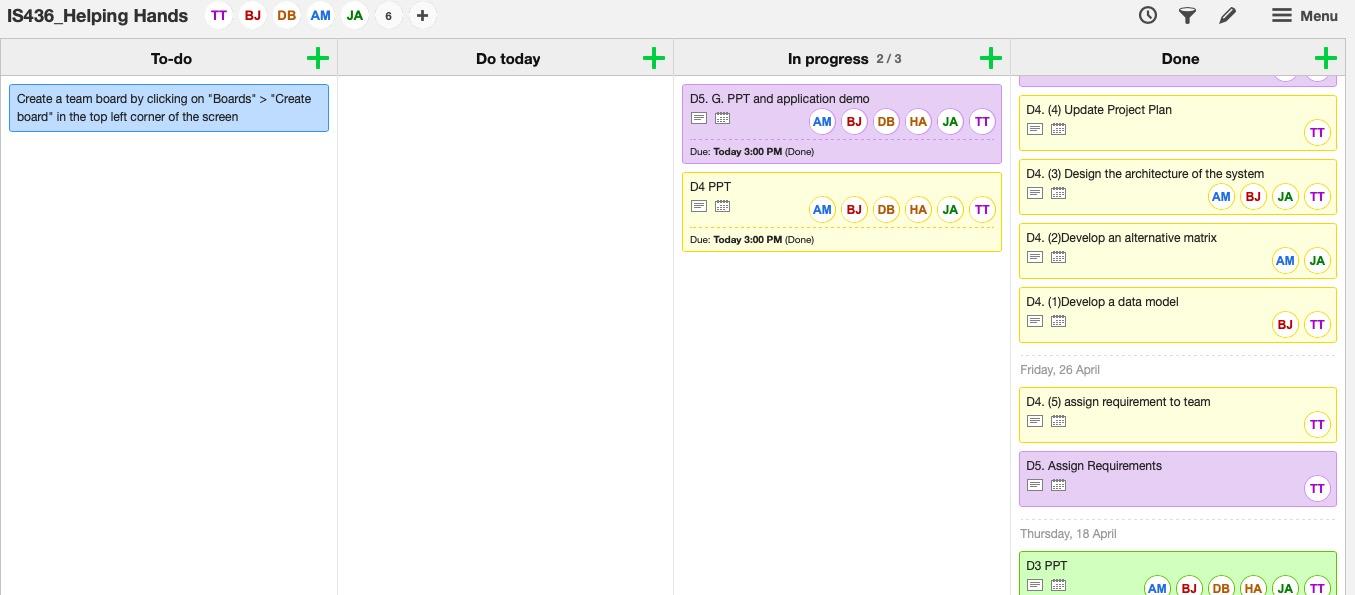
System Integration requirements will able to import and export data such as excel so the supervisor can generate reports in a easier way. The system will allow the supervisor to read and write appointments, as well as the database. Speed requirement is needed because our requirements stated that the system must update every 15 minutes. Availability and reliability requirements are checked because we need the app to function at all times. Access control because we only want the supervisor to use change certain features. Tutors can change their own schedule as well as the students.This will allow the supervisor to hand out punishments for student making no shows. Unstated Norms explicit will be used because our primary goal is to use it in North America as of the moment. We will be using date fields specifically such as date month and year in that order. Legal requirement is needed because

Our app is going to be used for tutoring purpose in a University, and it is important we must follow legal requirements for IT systems and be knowledgeable of any UMBC restrictions and legalities.

# Hardware/Software Specification

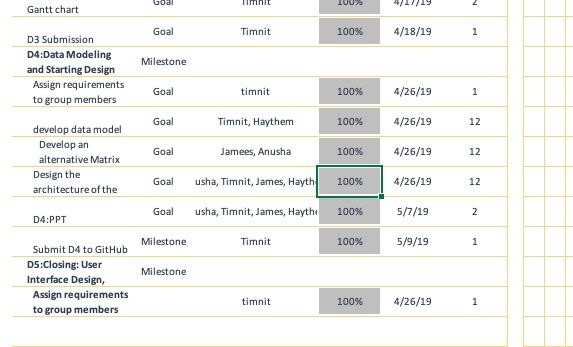
****

**KanbanFlow:**

****

**Updated Gantt chart:**

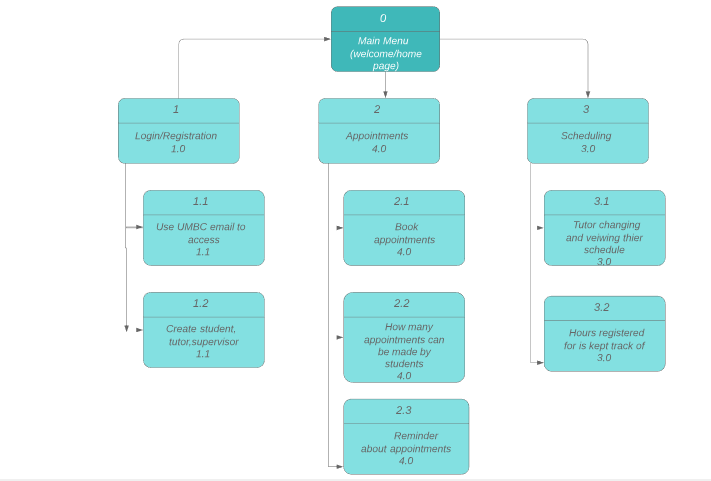
*(look for attachment:D4-Gantt chart\_Helping Hands)*



**Deliverable 5 – “Closing: User Interface Design, Program design and System**

**Implementation”**

**Interface Structure Diagram**

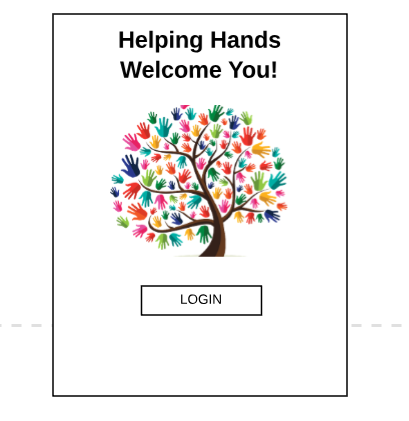


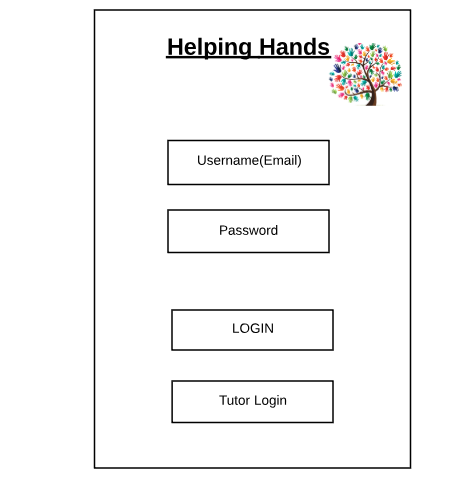
**Interface Standards**

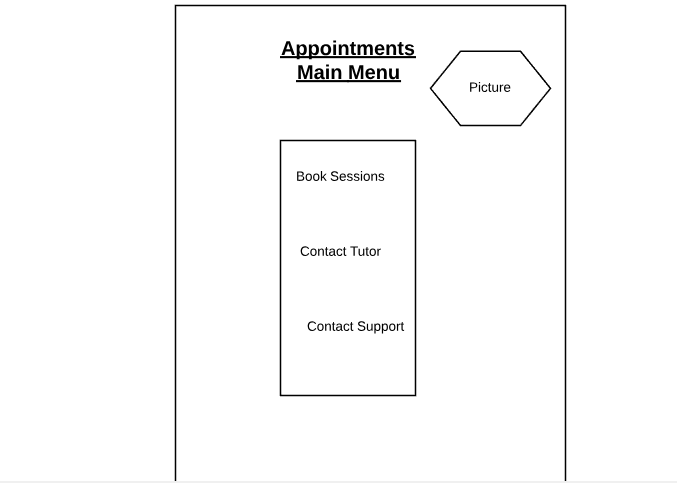
Interface standards are comprised of the basic design elements that are common across the individual screens, forms, and reports within the system. The following selected standards will ensure that the interfaces are consistent across the system.

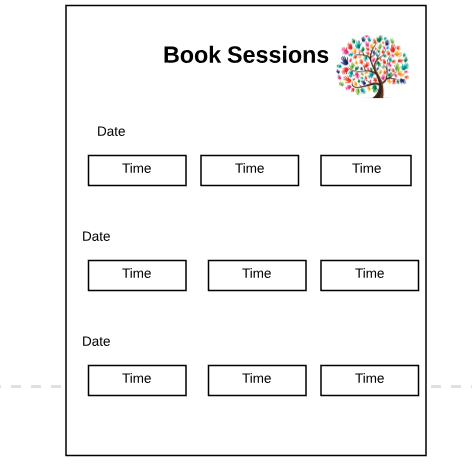
1. **Interface Metaphor** - a concept from the real world that is used as a model for the information system.
2. Login/ register metaphor: this will be used to symbolize access management software for the app
3. Calendar metaphor: this will be used to symbolize previous and future appointments that student and tutors register for.
4. **Interface Objects and Actions** - fundamental building blocks of the system such as entities and data stores (objects), and commands employed by users (actions). The objects and actions selected are all more general terms that would be simple enough for the end user to understand.
5. Objects:
   1. Entities: Students, Tutor, Supervisor
   2. Data Stores: Registration, and Scheduling Database
6. Actions:
   1. “Login” - allows all to get into the app
   2. “Booked Sessions” - allows for the tutor to see their appointments they have
   3. “Contact tutor” - allows the student to talk to their tutor when needed
   4. “Contact Support” - allows supervisor, tutor, or student to talk to support if something goes wrong
7. **Interface Icons** - interface objects, actions, and their status made to be meaningful, simple, and clear for the user.
8. Login button - the student, tutor, ad supervisor will be able to click it once their information is put in
9. **Interface Templates** - General appearance of all interface components in the information system. This will be used to tie together all of the other major interface design elements. Our team will be proposing the use of the app through appetize.io, which allows you to run the app on another browser. This particular software interface of the app consists of online appointment making and event scheduling between the student and tutor. The interface will combine all of the metaphors, icons, actions, and objects listed above. Most importantly, the interface must ensure user consistency throughout the system.

**Interface Prototypes**









System Requirements

*(to see code of app look at attachment of Code\_Helping Hands Folder)*

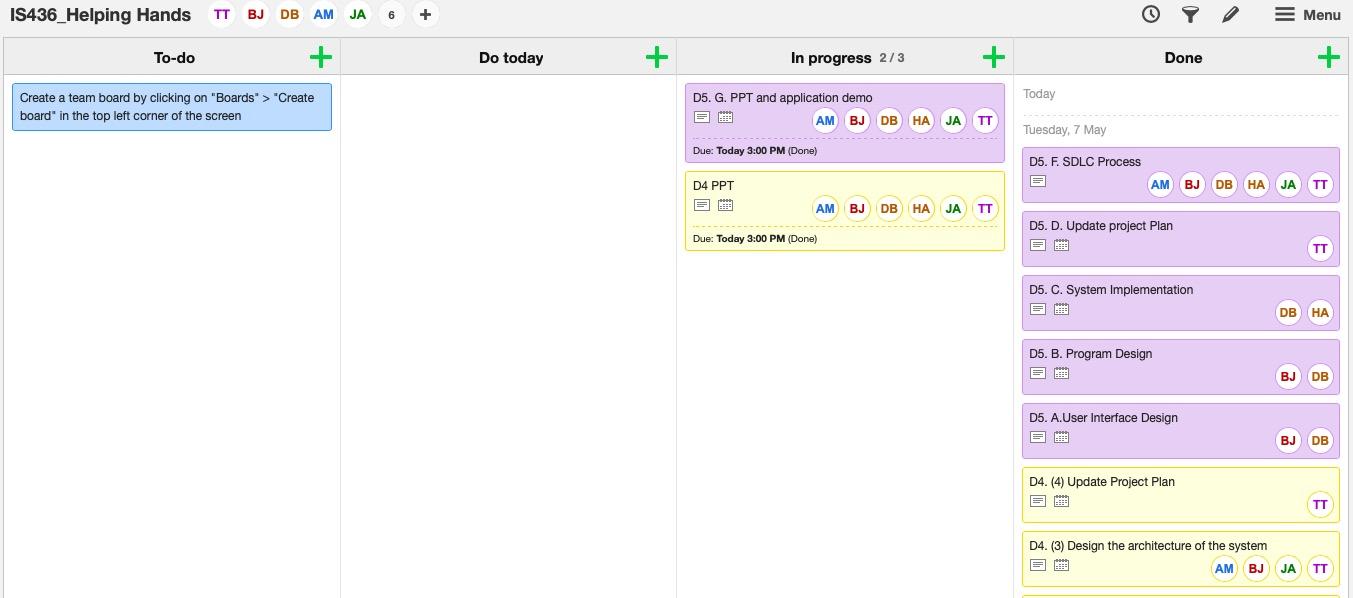
For implementing the app project, we used android studios and java as the programming language. From there we were able to design the interface design of each page and then coded the code that works between all pages. The hardware aspect was not too much used, but the software part was because the android studios software had to be downloaded but did not need any extras, like USB ports or SD cards.

The only requirements that are needed is a phone preferably a “smart phone.” Most “smartphones” have the necessary requirements to run an app and that is how our program is run. In addition, because it is an android app, it can only run on phones that are android.

Team Experience

The team experience throughout the SDLC process was great. We all were able to put our thoughts and heads together to come up with and solve the problem by going through the steps of SDLC. Also we analyzed the process it would take to finish the app and how it would be done. The designing part did not give us an issue, just figuring out which important screens we were going to focus on took a little time. Overall, the cycle helped the group because it helped us focus on one part of the project at a time. As a team, working together in this project has been very rewarding since we get to learn about the SDLC know how to work in an Agile environment.

KanbanFlow:



**Updated Gantt Chart:**

(look for attachment D5-Gantt chart\_Helping Hands)

