**UTDesign: Online Tutoring Application**

Team 63

Team Members:

* Jack Wittenbrook
* Mason Kuehne
* Chloe Lee
* Adithya Viswanathan
* Sam Salinas
* Vincent Tran-Bui

Abstract: For this project, we have created a web based tutoring application in order to help connect students and tutors in a simple and efficient way. To do this we have used a stack of modern technologies and coding languages including Angular, Django, MySQL and AWS hosting. By accomplishing this, students are able to find and book tutors with expertise on the exact courses they need help in and succeed academically.

### **Table of Contents**

[**Table of Contents 2**](#_nj161sxhvtn6)

[**Introduction 3**](#_pqkfogps03i1)

[**Timetable 3**](#_u1t1o43l0lhe)

[**Project Metrics 4**](#_wi6zwdtolc22)

[**Implementation Details 5**](#_qrxyaf88w4r8)

[**Impact, Security and Privacy 5**](#_d76pjdaues0w)

[**Individual Assessment 6**](#_5gydlmmgun55)

[**Issues and Lessons Learned 6**](#_4e1bslp202za)

[**Future Work 7**](#_9hdhtm9ok4s5)

[**Ethics Discussion 7**](#_fg622qwm8286)

[**Technology Stack 7**](#_x7kq0aihqba0)

[**Signatures 8**](#_bnl0llf7cmt7)

### 

### 

### **Introduction**

For this project, we have created a web based tutoring application in order to help connect students and tutors in a simple and efficient way. To do this we have used a stack of modern technologies and coding languages including Angular, Django, MySQL and Amazon Web Services. By accomplishing this, students are able to find and book tutors with expertise on the exact courses they need help in and succeed academically.

When conceptualizing our website development, simplicity takes center stage due to the busy schedules of both students and tutors. Recognizing that their educational commitments extend far beyond tutoring sessions, one of our primary focuses lies in ensuring efficiency and convenience in navigating our platform. Our project aims to provide a seamless experience for both tutors and students so that they can plan and view their sessions within seconds. Our priority is intuitiveness and usability. The final application should be user-friendly, with little room for confusion. Layout of components will be simple but intuitive.  
 One aspect makes our app unique is the fact that it specializes in instruction for courses taught at The University of Texas at Dallas. While most common tutoring apps have a wide variety of subjects that fall under math and science, our app takes it a step further by matching the exact course rather than having to hope that the tutor remembers the specifics. For example, if a student is struggling in CS 2337, they will be able to find a tutor that is knowledgeable in that exact course and who understands the curriculum being taught in that class. Students can simply log into the website, search up their exact course title, and easily find help with the class that they are looking for. Tutors benefit from using our app because they will not have to worry about discrepancies between similar courses at different schools.

A similar solution is UTD’s Peer Tutoring Service. Sophisticated and useful for students, this service gives extensive opportunities to students if they need assistance with any UTD-related issue. Although our application is more lightweight and not as feature-packed, it is likely to be more user-friendly and overall simpler to use, from a user’s perspective. Our menus are less cluttered, our options are trimmed and comprehensible, and most importantly, our platform can easily be built upon. The simple but powerful tools such as Angular, Django and Amazon Web Services allow for immense scale and growth.  
 Ultimately, we want both students and teachers to be on our website for as short of a time as possible, while finishing the task they set out to do. By providing user-friendly navigation and clear information about available courses, we aim to achieve this objective and make the educational journey smoother and more efficient for all parties involved.

### **Timetable**

For our development we have used 1 week agile sprints in order to keep a reasonable structure and timeline. Each week has a general deadline to keep the progress on track but also allow the schedule to be flexible and handle delayed or early tasks. During our weekly team meeting, we review what has been done over the sprint, plan out what tasks will need to be done for the following week and discuss any issues that others might be able to resolve or help with.

Below is our general time schedule set at the begining with the base objective that was at least mostly accomplished that week. Anticipating setbacks, the schedule has several weeks toward the end of the project to allow for some flexibility to catch up, polish the finished application and work on the final materials.

| Sprint # | Dates | Base Objective |
| --- | --- | --- |
| 1 | 10/07-10/13 | Set up project components |
| 2 | 10/14-10/20 | Login/Authentication |
| 3 | 10/21-10/27 | Tutor Profile |
| 4 | 10/28-11/03 | Search Tutors |
| 5 | 11/04-11/10 | View appointments |
| 6 | 11/11-11/17 | Book appointments |
| 7 | 11/18-11/19 | Polish and finalize |
| 8 | 11/20-11/26 | Fall break, finalize project |
| 9 | 11/27-12/01 | Finalize project, project report and presentation |
| 10 | 12/02-12/07 | Finish final report, presentation prep |

### **Project Metrics**

One basic form of metrics to gauge the success of the project is a survey or form asking users (students and tutors) to rate their experience with the website. Questions will be geared towards these main categories: ease of use, design, loading times, and feedback. These questions will be formed and asked as unbiased, non-leading questions. This will help avoid biased answers and will allow users to express their true feedback with the user experience of the application. The questions will be similar, but not identical, for students and tutors. Due to their difference in experience, feature set, and overall role towards the application, the questions will be tailored to their experience. Once the users have spent significant time on the platform, a feedback form will be sent out (likely to their email address) for them to fill out. This will be collected and stored in the database for further analysis.

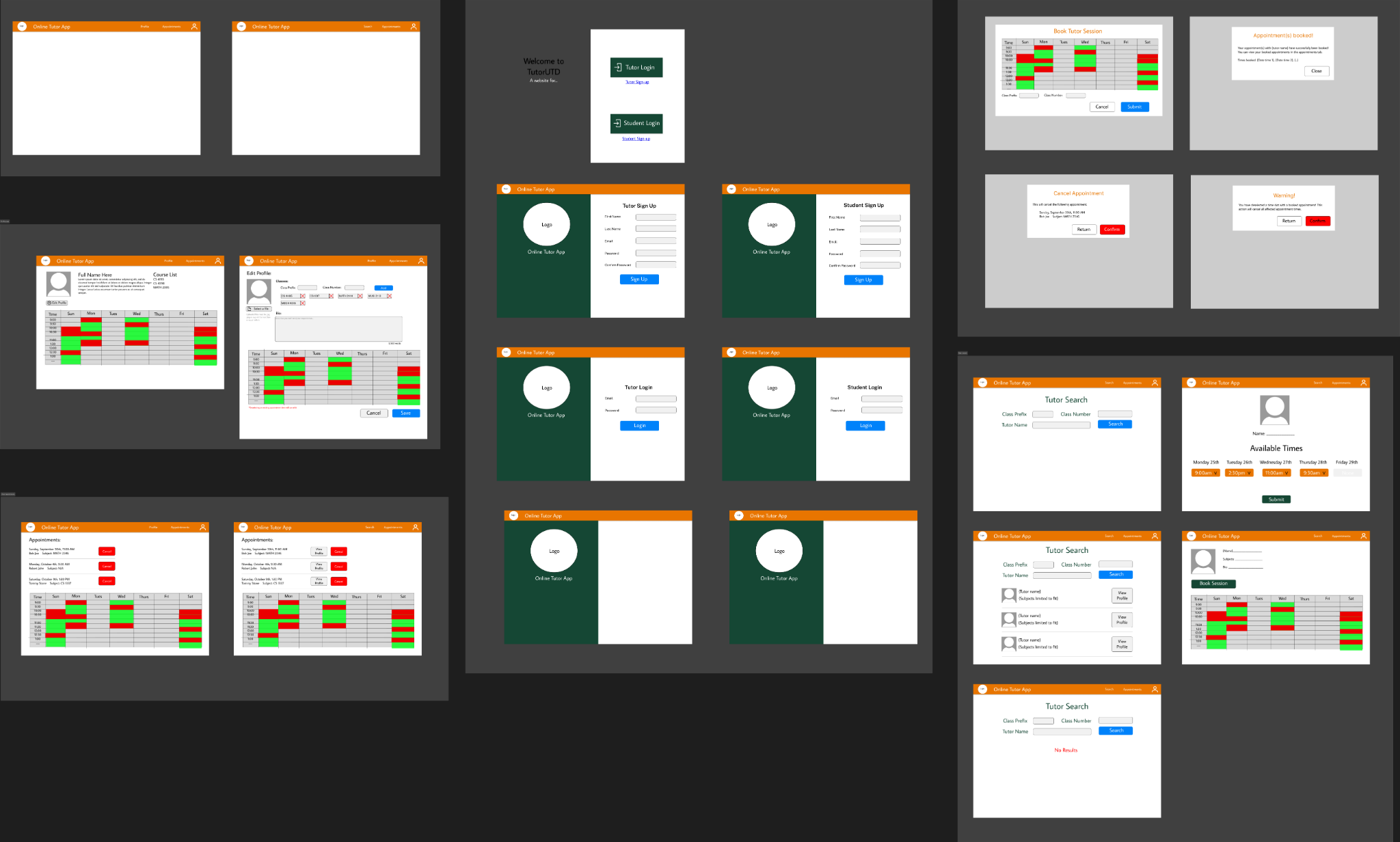
Another metric that will be tracked is the total time spent in tutoring sessions by both tutors and students. The system will calculate and add to each user’s total when either the tutor or student marks an appointment being complete, which adds the appointment time to both users and marks the appointment as complete in the database. This can be analyzed by us to find the users making the most use of the service and look for patterns or ask for feedback. This can additionally be viewed and used by users to gauge their own use as well as be viewed by students that might have more confidence in seeing tutors with high amounts of hours spent tutoring.

As this project has not had any public use or user feedback, we are unable to have any meaningful numbers for these metrics. However once users are able to use the application the existing systems and functions will be able to collect and track these statistics for further use.

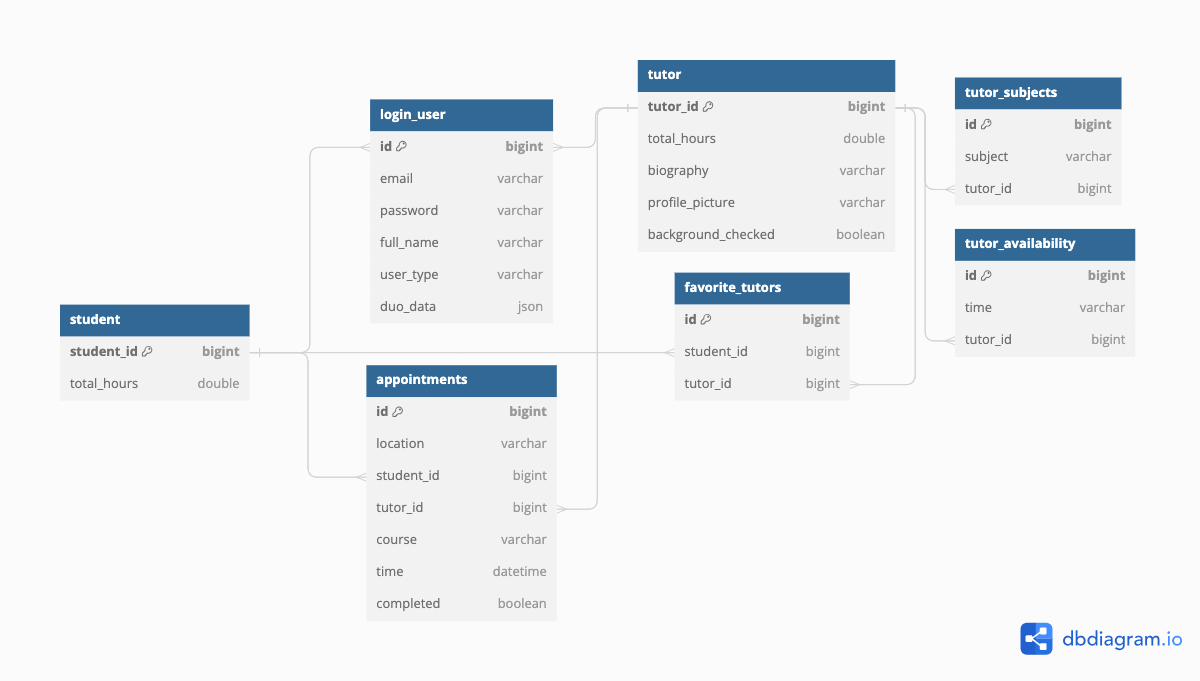
### **Implementation Details**

Source Code: <https://github.com/AdithyaViswanathan1/TutorUTD>

### Figma Design



### Database Schema



### Backend API Documentation (For Major Components):

| Student/Tutor REGISTRATION  Route: /login/{student/tutor}\_register  Input: Full Name, Email, Password  Output: Returns a link of a QR code to enroll users into Duo for authentication purposes and HTTP status created. A confirmation email is sent. | Student/Tutor LOGIN  Route: /login/{student/tutor}\_login  Input: Email and password  Output: Returns a link of a QR code if users have not enrolled into Duo or user ID and HTTP status OK or HTTP status unauthorized |
| --- | --- |
| Tutor GET PROFILE  Route: /tutor/profile  Input: JSON Request containing token   | {  "id": 17 } | | --- |   Output: JSON containing attributes of tutor   | {  "tutor\_id": 4,  "user": {  "full\_name": "adi 04"  },  "total\_hours": 0,  "subject\_list": null,  "biography": null,  "hours": null,  "profile\_picture": null,  "background\_checked": false,  "available": false } | | --- | | Tutor EDIT PROFILE  Route: /tutor/edit\_profile  Input: JSON containing tutor\_id and fields needed to be updated   | {  "tutor\_id": 17,  "biography: "Hi, bio changed",  "subject\_list": ["cs 101"] } | | --- |   Output: Success/Fail message + HTTP status |
| Student/Tutor CANCEL APPOINTMENT  Route: /{student/tutor}/cancel\_appointment  Input: ID of appointment to cancel   | {  "appointment\_id": 6 } | | --- |   Output: HTTP status OK/BAD. Successful cancellation sends confirmation email to both student and tutor. | Student/Tutor MARKING APPT. COMPLETE  Route: /{student/tutor}/mark\_app\_as\_complete  Input: appointment id   | {  "appointment\_id": 6 } | | --- |   Output: HTTP status OK/BAD. In the background, marks appointment as complete and adds 0.5 hours to student and tutor fields. |
| Student SEARCH FOR TUTOR  Route: /student/tutor\_search/  Input: JSON containing fields you want to search by (can search by course\_prefix and/or course\_number, or just tutor\_name)   | {  "course\_prefix": "CS",  "course\_number": 17 } | | --- |   Output: Returns list of dictionaries. each dictionary corresponds to tutor object containing the Fields: id, full\_name, subjects, profile pic + http status   | "Result":  [  {  "id": 21,  "full\_name": adi 08,  "subjects": ["cs 17",  "cs 18"]  },  {  "id": 22,  "full\_name": adi 09,  "subjects": ["cs 909",  "math 18",  "cs 17"]  }  ] | | --- | | Student/Tutor GET APPOINTMENTS  Route: /{student/tutor}/get\_appointments  Input: Student/Tutor ID   | {  "appointment\_id": 54 } | | --- |   Output: List of appointments for that student/tutor (now filters to non-completed sessions)   | [  {  "id": 5,  "student\_id": 54,  "tutor\_id": 58,  "time":  "2023-12-02T03:00:00",  "location": "JSOM 1232",  "course": "MUSI 1300",  "completed": false  },  {  "id": 6,  "student\_id": 54,  "tutor\_id": 59,  "time":  "2023-11-29T02:00:00",  "location": "ECSS 1909",  "course": "MATH 1023"  "completed": false  },  {  "id": 7,  "student\_id": 54,  "tutor\_id": 59,  "time":  "2023-11-29T11:00:00",  "location": "ECSS 1909",  "course": "MATH 1023"  "completed": false  } ] | | --- | |
| Student MAKE APPOINTMENTS  Route: /student/make\_appointment  Input: A JSON dict of containing “student\_id”, the “tutor\_id”, the “dates” which can be multiple dates that the student wants, and the “location” which can be online or at a specific location.    Output: Returns HTTP 201 CREATED on successful creation, and 400 BAD REQUEST on failure. Sends confirmation email to both tutor and student      Constraints: The date must be a time in the future. | System DailyEmailReminders  Route: /online\_tutoring/cron/dailyEmailReminders  Input: N/A  Output: No return. Email is sent to tutors and students  Description: Sends emails to tutors and students with appointment details for the day |

### **Impact, Security and Privacy**

This project will help connect students and tutors in a simple and efficient way. By implementing this project, struggling students will be able to easily get the help they need to understand their classes by connecting with the needed tutor(s). Compared to some other similar resources, we will focus on the college environment. Particularly, the search function will allow students to search for specific classes so that students can find a tutor with the exact specialty they need rather than just the general subject. While the other approach may suffice for the main path classes that any tutor would have been in, students would find difficulty in seeking help for the more niche elective classes. With our searching approach, we can serve the needs of all students and classes more equally to ensure each student can find the perfect tutor for them.

For the security of our application, all users must login with their email and password in addition to using Duo two-factor authentication before being granted access to the site. With two factor authentication, we can prevent spam and bots from attacking the site for an improved user experience. During account creation the only data we collect is the user’s name, email and chosen password. We will be hashing the user’s password using SHA-256 protocol to ensure that the password can not be directly read by any unauthorized parties. Another security measure within our application is background checking. Only when a tutor is background checked, and a database administrator has changed their status, are they listed for students.

### **Individual Assessment**

**Jack:** For this project I worked generally across the front end. For the first weeks, I planned and researched the structure and technologies to be used on the project. I had some full stack experience from an internship and tried to incorporate those technologies such as Angular but still had to learn the connecting parts of the application and the initial setup. Also in the first weeks I did much of the figma designing to plan out both the design and to think through the functionality and needs of each component in the final application. Once development started I handled the initial setup and organization of the front end of the project and added useful packages such as font awesome and bootstrap to help development. After that I implemented the design and core functionality of around half of the front end components and pages including the landing page, sign-up page, time-table component, appointments page, search page, favorites page, duo enrollment page and several pop–up modals. In addition to creating these pages and components I linked the created backend routes to the front end by setting up the necessary front end services and http manager components and routing the data appropriately to the previously created components. I also implemented route-guarding using cookies to ensure that unauthorized users could not directly navigate to inner pages and that logged in user’s could skip past the login page as well as ensuring student and tutor users do not end up on the wrong versions of pages. Finally, in the last couple weeks of the development time I have primarily finished the last http calls, implemented minor features such as loading icons, fixed various found issues and worked on the final deliverables. While the front end part I was mostly working with was not covered by any of my previous classes, I found CS 3354 Software Engineering helpful especially as the leader in thinking through, planning and managing the project. As my suggestion for the course, I would say for the UTD sponsored projects in particular to give some resources to help with the initial learning curve since many aspects of the projects are not covered by previous courses at all.

**Mason:** My role during this project was to assist with frontend development. After seeing the initial design, I wanted to begin work on the navbar component because it was universal across all other components. Implementing the navbar and the conditional variables alongside it (who was viewing the website, what screen is the user on) took about a week. After the navbar, I decided to work on the profile component. It took a few weeks of work to sort out where everything would go on the page, how we would access different variables, how the tutor would be able to edit their profile, and how things would change on the student's view of the profile. Several modals were required to complete the profile portion of the website, specifically for editing the profile and booking a session with a tutor. I spent the next couple of weeks focused on redesigning the website. The frontend was almost all good to go at this point, so we decided we wanted to fill up some whitespace and make the website look more consistent across the different pages. We found a new color palette that we all agreed on and also added some depth to the website through the use of shadows. Finally, I assisted with the final documentation.

**Chloe:** I was a part of the backend development. The first several weeks of development I spent researching and learning about Django, the REST framework, and Python as a language. After the learning and researching, I began to implement the login and registration section of tutors and students. This implementation started around October 14th and lasted until the 26th. In the following month of November, I started to implement authentication with Duo and completed the implementation on the 8th. The registration/authentication sprint was supposed to last a week (from October 14th to 20th) but it proved more difficult than expected and unfortunately the planned deadline was not met. Afterwards, I focused most of my attention to minor tasks (cleaning code, running the program for errors, debugging, etc.) and paperwork (presentation, script, and final documentation).

**Sam:** I worked on the backend of the project. During the first few weeks of planning I spent time researching Django and re-familiarizing myself with AWS. I also spent time designing the workflow diagrams for students as well as designing the different sequence diagrams that the backend would need to refer to. From there, I moved on to learning Django. During my time learning Django, I spent time learning about the features that Django has to offer by watching videos which were tiny projects to familiarize myself with the tool. As I got familiar with Django, I began to look on AWS for the resources that we would need for the project, such as the database. While we never officially kicked off the project onto an actual EC2 instance unfortunately, we did plan out the design for the servers by separating the frontend of the project and the backend of the project onto separate instances. Once I got comfortable with Django, I kicked the backend off by designing the structure of the project by creating the different routes that the team would be using, such as the /student/ route and the /tutor/ route, which was just a template to help the team get moving. After some login design from Chloe, I helped fix any bugs that we encountered with login and then moved on to designing part of the Student route. I designed most of the features regarding appointments, as well as students being able to mark their favorite tutors, with the exception of retrieving appointments being designed by Adithya, which we coordinated with the front end on.

**Vincent:** I focused on the backend of the project. I initially started off acquainting myself with the frameworks and technology we were using as I had no prior experience using Django and its REST framework. At the same time, I fleshed out the student workflow diagram during our planning phase. When development started, I began working on the authentication routes of the application. Quickly though, we recognized the need to move our routes from Django to the REST framework, which greatly prolonged the development time of the core feature. Afterwards, I began looking into the appointments routes and then focused my attention on the email additional feature. I implemented the email routes but also needed to consider what email service and account system to use. I decided to create a mock gmail account that sends through Google’s SMTP servers as it would most closely represent how UTD would have a custom made email account that will send through the Outlook SMTP servers. Finally, I worked on the final documentation.

**Adithya:** In the first several weeks of the semester, I worked on designing the database, including researching which type of database to use, which platform to host it on, what architecture to use based on the requirements, and how to connect it to our backend Django framework. After we came up with a plan, design, and implementation ideas, I started learning extensively about Django. This period lasted from the last few weeks leading up to the mid-semester project proposal and a few weeks into the development phase as well. More specifically, I learnt about Django REST Framework and how to build REST APIs. This required knowledge about handling requests from the frontend, writing code to handle it, and if required, make changes to the database (on AWS) accordingly. Using the skills I learnt from numerous tutorials and a small side project, I became proficient in Django backend development. My first contribution was to the login/register functionality in the first few weeks of the development timeline. Then, I implemented most, if not all, of the tutor functionality. This includes tutor profile CRUD operations (i.e. profile picture, biography, subjects they teach, booking tutor appointments, available times, etc). This spanned from the 3rd or 4th week of development till the end of November. In the fall break, I implemented the student’s search engine for tutors with keyword filters such as course name, course number, and tutor name. Finally, the week leading up to presentation delivery and submission, I worked on testing our application for bugs and fixes. I also worked on the presentation material itself (slide deck, script, and documentation).

### **Issues and Lessons Learned**

**Jack:** During development, I did not have too many notable issues with the coding itself outside of normal errors and tricky parts largely due to spending most of the summer working on very similar code that I designed much of this project’s structure and techs to be similar to. The main development issue I encountered was mismatched progress between the front and back ends with the http routes not being ready at the same pace as the front end pages that use them which resulted in having to use dummy data for a long time and backtrack when the routes were ready. I think this is more a result of the learning curves than individual issues since Mason and I both have experience with Angular front-end development while the back-end side was mostly learning from scratch. When linking up the http routes especially on the early end I had a difficult time trying to figure out the right formats to send and receive in due to differences in format from different people writing the routes as well as not understanding how the Django code worked, making me rely on the back-end people having to explain the formats and names they were expecting. A lesson I learned more outside of development was even in the leadership position to be less stubborn in my vision of the project, where in the planning stages especially I was more insistent on trying to recreate the structure I had worked with before rather than adapting to what might suit the others better and dismissing ideas that didn’t quite fit my vision of how the project should function. Over the course of the project I feel that I have become more flexible and communicative with my team members to achieve a more shared perspective and allow others to make better use of their strengths as well.

**Mason:** The biggest issues I had while working on this project stemmed from a lack of communication with other developers. While we did have a discord chat that was used frequently, I personally had a hard time catching up with changes on the frontend. If I couldn't figure things out on my own, I would typically ask Jack since he was the other frontend developer. After taking a bit of time each week to catch up on what everyone had done and the new libraries that were implemented, my own personal development went pretty smoothly. It was difficult at times to know what to work on because the rate at which the frontend and backend advanced were inconsistent at times, meaning that I would have to sometimes use dummy data as opposed to properly linking to the database. If I were to start this project again, I would want to have each developer be full stack and sort work by components as opposed to frontend or backend. A big lesson I learned while developing the website was how to communicate between child and parent components. For example, the edit profile modal required information from the profile component, which got its information from the backend. While I did have some previous frontend experience before this project, working on the app really strengthened my understanding of the parent/child subject. I had some issues later on in the project's development once more work got done in the backend. It took me a while to install necessary libraries and get the backend running, but that was mostly because of my lack of experience of manually installing individual libraries.

**Chloe:** Throughout the project, I had many issues primarily being the steep learning curve. I was largely unfamiliar with Django (along with the REST framework), web development, and Python. Despite researching for several weeks, practicing Python, and following the Django basic project guidelines, it still proved difficult to code the login/registration aspect. However, I did overcome it as I continued to research throughout implementation. I read through multiple helpful resources from Django’s tutorials to guides to official code documentation and compiled the information into tools that I could use and incorporate into my thinking. Unfortunately, another issue with the login/registration was that I was unable to put users into the database. As such, I realized rather than antagonize over something I did not understand, I shared the problem with my backend teammates and managed to resolve the issue. As I got used to the platform and the coding language, other implementations, the Duo two factor authentication, went relatively smoothly. The only issue I had was that I believed I could use the Django-Duo package and tried to incorporate that into the project. Ultimately, I believed that using the package was infeasible as I ran into error messages and could not get it running properly. As such, I decided to code it manually using the Duo API which made it easy to spot the errors and fix them. From previous courses, I learned that it is easier to code in small sections rather than large sections and from this term I learned when to ask for help. These lessons aided me greatly throughout this project as I continuously made ‘baby steps’ and slowly implemented one aspect at a time, confirming that it worked, then implementing another aspect. I also learned when to ask for help when needed rather than to antagonize over a problem myself and completely halt progress. At the end of the project, I gained a deeper understanding of web development and learned how to research and compile the relevant information necessary to complete a task. The senior design course was a good lesson on how to adapt to change and think ‘outside of the box’ as I was completely unfamiliar with the topics addressed in the project. I am unsure how to improve the course further.

**Sam:** The first issue that was noticeable especially during development time was the time it took to get acquainted with Django. Django had a steeper learning curve than initially thought, but once I went past the hurdle, development became much quicker on my end. On top of that, I had too much of a focus on security for the project. Whilst it is necessary for the application to be completely secure, we are currently not dealing with extremely sensitive data, with the only data being managed are the first name, last name, and email. I realized this somewhat late, but it definitely slowed down my development time trying to idealize a secure web app when my first goal should be to get a working product. With that being said, there were a lot of ambitions that I had for this project but I was not able to follow through on which I had talked about in our meetings, and I need to get a working product out ASAP before I focus on that.

**Vincent:** The issues I encountered during this project mostly stemmed from my lack of experience with Django and our other frameworks. While developing the authentication services, two issues arose. The first was a miss ordering of tasks, where the establishment of user objects should have come first, otherwise the authentication routes would have nothing to interact with. The second was realizing that Django alone was not enough to create the application we intend. It was during that time that we found and introduced Django’s REST framework and a significant amount of time needed to be spent figuring out how the framework worked and retooling our existing code to work with it. While developing the email features, the main issue I encountered was weighing what email service to use. The options available were to use our personal utd email, gmail, or other third party services like SendMail. Using our utd email addresses would be out of the question as that would require putting our personal passwords in a file. Services like SendMail are designed for use across many applications and scale well to high volume email sending. If our application was not geared for use by UTD, SendMail would have been the likely choice but gmail was chosen to imitate the way UTD would likely implement email services whilst also providing an example of a non-UTD implementation. From these issues came the most important lesson I learned, the importance of project management. All our inexperience setting up projects, working with other people, and maintaining a schedule, contributed to more issues and disorganization during development. I definitely feel like we underutilized and underestimated the utility of the planning phase of development. Moreover, during the planning phase, I sought to get into coding as soon as possible but that clearly caused issues. Given the opportunity for a do-over, I would make far more in-depth planning materials and use that to well organize project tasks and speed up development on individual parts. I also learned to ask for help and how useful it is to get a fresh pair of eyes on code to point out the smallest mistake that's been overlooked. Some suggestions on improving this course would be to provide general project management guidelines and resources to make projects go by smoother.

**Adithya:** Overall, my contribution to the design and development of TutorUTD was filled with memorable lessons and moments of learning. One lesson I learnt from my previous courses (most notably database systems and UNIX programming) was to have a solid plan for implementation before diving into coding. This is only through good design principles. The original database schema that I planned gave me a thorough idea of all the pieces I had to fit together in the backend development. The schema was also influenced heavily from the concepts that I learnt from database systems in my junior year. ER diagrams from my database course taught me the different parts of database design, such as entities, relations, attributes, etc. Although it changed from my initial conception, it provided a useful baseline to build upon. During the midpoint in the semester, after I had finished designing the database and finished thinking of how to fit the database and backend together, I was at a crossroads on what to pursue for development. I knew that after implementing the database, I would have to take up either the front-end or back-end. But, in both cases, I had no prior experience or even an ounce of understanding on how to start. Therefore, I learnt that I must go out of my comfort zone and spend quality time on learning about Django (our backend framework). Through a week full of tutorials, analyzing different projects, I gained an understanding of how REST APIs worked. And from there, I was able to contribute more than I imagined to the backend. Overall, this experience taught me one major item: to value uncertainty, new experiences, and fearlessness to seek outside my comfort zone. I should welcome change throughout the process. The efforts and proposals we present in the beginning will ALWAYS change. I learned that this is natural and we should adapt as we go along; since we learn and grow with the project, so will our ideas, our implementation, and ultimately, the end product. Overall, I am grateful for this senior design course. If I could change one thing, it would be the duration of the development phase. I would make the development phase a bit longer to accommodate for more coding time. This will likely help the implementation of a few more features or aid in testing the application more completely.

### **Future Work**

With the given time, only a limited amount of features can be implemented. Here are ideas for further improvement of our application.

1. Mobile-friendly platform
2. Refined website images, design, and theme
3. Integration with online meeting platforms (Zoom, Google Meet)
4. Feedback form (through email) to collect user experience information in order to improve application in future iterations.

### **Ethics Discussion**

Maintaining moral and ethical boundaries can be achieved in many ways. First, it is important for each team member to recognize their roles and responsibilities. It is also important to make sure the work is split evenly across all team members. One way this can be done is to estimate the amount of time and effort needed to complete a set of tasks based on a member’s skill level. If that team member chooses to be held accountable for their work and the team comes to a consensus on the responsibilities needed to complete that set of tasks, then the task can be assigned.

Additionally, the nature of the project will likely require exploring similar open-source implementations online. The ethical discussion of plagiarism can be handled in the following way. Other repositories cannot be directly copy and pasted into our project but can be used as examples which may inform us to pursue a certain direction, use a specific service, or implement a component in a particular way. Our project should never mirror an example or use it directly.

In the end, a major source of information for any implementation is documentation from a respective platform website or design website. For this project, this includes:

* https://angular.io/docs
* https://docs.djangoproject.com/en/5.0/
* https://dev.mysql.com/doc/
* https://www.figma.com/
* https://dbdiagram.io/
* https://docs.aws.amazon.com/
* https://duo.com/docs

### **Technology Stack**

Front end:HTML Typescript, SCSS

Back end: Python (Django)

Database: MySql

Hosting: AWS Free tier

2FA: Duo

Designing: Figma

Additional Frameworks and libraries: Angular, RXJS

### **Signatures**

| **Jack Wittenbrook (12/7/2023)** | **Adithya Viswanathan (12/7/2023)** |
| --- | --- |
| **Chloe Lee (12/7/2023)** | **Mason Kuehne (12/7/2023)** |
| **Sam Salinas (12/7/2023)** | **Vincent Tran-Bui (12/7/2023)** |
|  |  |