Online Typing Application Report

COMP8045

- - – A00000000

1-14-2020

Table of Contents

1. Introduction

1.1. Student Background

My name is - -. I graduated from North Delta Senior Secondary in 1998, then went to Kwantlen University College for two years. Before I found an IT job, I worked as a sales marketer, warehouse shipper receiver, customer service in different companies from 2003 to 2012. I decided to change my career in 2012, because I want a challenging career with great potential to improve myself. I quit my previous chemistry lab job to attend BCIT CST program in September 2013 as a full-time student. After CST first year, I found an eight months IT-Service position in Boeing Canada in BCIT's Co-op program. After Co-op, I finished the second year CST courses to get my diploma at the end of 2015. Then I found a fulltime software support and developer job in Schneider Electric in Burnaby. This is the first job I found in IT industry. I support and develop Schneider's internal web applications and databases. I also maintain data in ERP (Enterprise Resource Planning) systems. Since September 2016, I started my BTECH program as a part time student. I am in both wireless and mobile and human computer interaction major. Now I only have this project left to get the bachelor's degree.

1.2. Project Description

This project is for BCIT BTECH's project course COMP8045. This is my individual project from proposal, design, implementation, test and report.

The goal of this project is to create a free online typing software to let users, especially elementary school students to practice their typing skills. This project will mainly consist of a web site client which can be access from both PC and mobile devices, a Microsoft .Net Core web server and an SQLite database.

1.2.1. Essential Problems

The essential problem I solved is to improve typing learning efficiency by collecting errors users made during practice. Let me to use a body building workout example to further explain the essential problem of current typing application.

The following is a workout schedule for body builders.

15 pull ups

20 barbell shrugs

20 bench press

20 push ups

40 barbell squats

30 minutes cardio

Do it every Monday Thursday and Saturday.

Let's assume some people real like this schedule and it really works for them, but will this schedule be efficient for everybody? Definitely no. That's why a lot of bodybuilders hire professional trainers to customize unique training programs for them based on their needs and conditions. For instance, Tom's legs are relatively weak, so he should do more squats. Joe likes big chest, so he should do more bench press etc.

Back to our typing learning. When we open a typing software, if all learners will use all the same sample texts to learn typing, it's like all bodybuilders use the same training program to workout. Many people hire a personal workout trainer, but who likes to hire a personal typing trainer to guide them to use the most efficient practice to increase their typing proficiency? That's why many typing learners only use the same predefined texts to practice typing. They waste a lot of time on typing something they are already very good at, but do not have chance to practice what they are not good at. The application I am going to develop is the personal typing trainer that has the unique feature in the world for all learners and it is free. My application will measure users typing mistakes to see which keys they have higher chance to type wrong

1.2.2. Goals and Objectives

The goal and objective of this project is to create a free online typing application, so that registered users can save their typing result, then they will get suggested typing practice according to their previous typing mistakes. The objective is to find a more efficient way of learning typing and prove that focus on correcting mistakes during practice for each learner is more efficient than letting all learners use the same practice.

First, my typing application will provide features that current typing software do not have. My design is based on my typing and language learning rational which is to combine typing with learning vocabulary, grammar and writing skills. My typing training application will let users customize what they use to practice typing, for example, save selected text to practice. It is potentially can be used by elementary school teachers to design typing practice for students, so students can use typing to learning spelling, grammar etc. at the same time.

Second and more importantly, unlike other online typing applications only provide users very general type error reports, my typing application can suggest users what they should practice more according to their recent typing report. (The detail of the algorithm is in Technical Challenge section). Each individual learner will get different suggestions which target their specific weaknesses in typing. Teachers can save many sample essays into database, then my typing application has an algorithm to analyze the typing errors for each student and then select previously saved essays to suggest students to practice. If a student often makes mistakes about key "w" and "o", then either words with more "w" and "o" are suggested to this student to practice, or a pre-uploaded essay which has more "w" and "o" are suggested. This is another education rational of mine. Currently, students learn the same thing by using the same practice handed out by teachers. I believe that even though we learn the same skill and there are common learning principles, everybody's practice path will not and should not be exactly the same, because everybody is unique in this world. Everyone has their own strengths and weaknesses, so ideally everyone's practice should be customized based on the observation and measurement of their unique condition. Beyond the technical challenges for software development, beyond typing skill, this project is also my personal preliminary exploration about how to make education more efficient. This typing application is also a test tool for my

education rational. Of course, the current technologies in our social education system are not able to make different targeted plans and exercises for each student's learning in every subject, but I believe in the future it will become possible after internet, big data and artificial intelligence are further developed.

2. Body

2.1. Background

I found this topic is useful and interesting. There is no company to be my sponsor. The main idea of this problem came from the observation of my daughter's typing practice. In each of my daughter's typing practice, I saw her repeated what she had typed correctly many times and still made the same mistake as her previous practices. Then I thought she should spend more time to focus on practicing her previous mistakes, so it is critical to collect and save her typing mistakes for further analysis. Her future practice should be selected according to her previous typing records.

I searched on internet, but all typing software only have pre-selected static practices. I think I can create a new typing application which can collect users' typing records, then suggest the best typing practices according to users' previous typing mistakes to improve learning efficiency.

2.2. Project Statement

The typing training application should be able to analyze user's typing practice result and suggest user the best practice to increase learning efficiency.

2.3. Possible Alternative Solutions

A possible alternative solution is to use LAMP which is Linux, Apache web server, MySQL and PHP to develop this web application. I used LAMP to develop an application in my CST group project course. The advantage of LAMP is that all components of developing tools are open source and free. Moreover, these are all mature technologies and there are a lot of help documents online.

2.4. Chosen Solution

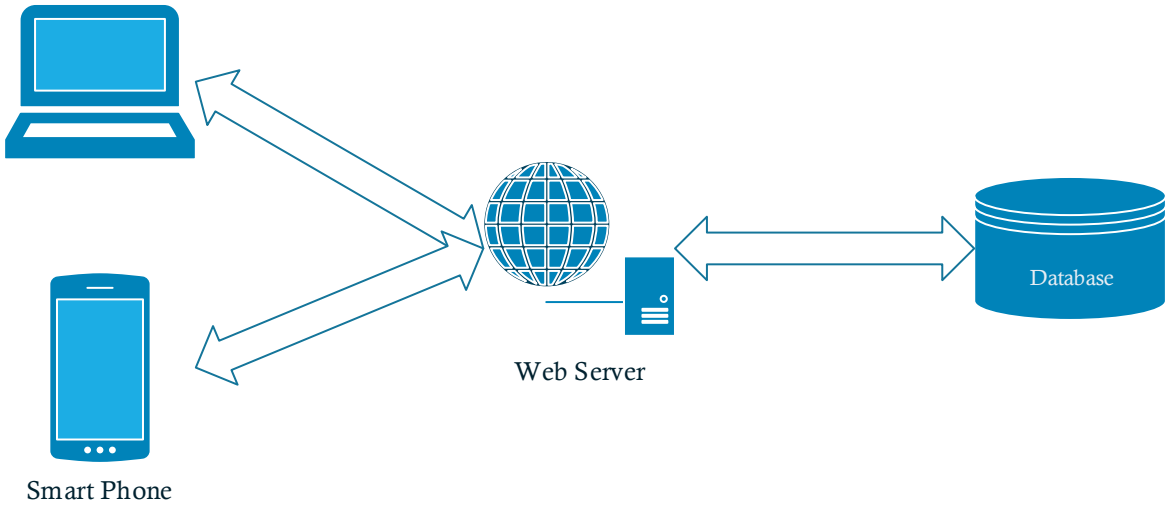
My chosen solution is to use Angular 7 as client, ASP.NET Core Web API as web server, and SQLite as database. I think this is a better solution compared with LAMP. First, the Angular, ASP.NET Core and SQLite are also open source and free. They are also mature technologies. It's easy to find help online too, so it makes LAMP have no advantage compared with Angular, ASP.NET Core and SQLite in this application. Moreover, ASP.NET Core is cross platform, which means the web server can be hosted on Linux, Mac and Windows, but LAMP does not work on Windows. Furthermore, SQLite is a small files-based database, so it can save a lot of database server maintenance work. Therefore, I believe Angular, ASP.NET Core and SQLite is a better solution for my online typing application.

2.5. Details of Design and Development

2.5.1 Context and data flow diagram

This application consists of three components.

- Web application client: Angular web front-end that users use to practice typing and see results. User requests will be sent to web server.
- Web server: ASP.NET Core Web API receives, processes and responses user requests.
- Database: SQLite stores application and user data permanently. Web server can query, save, delete and update database.

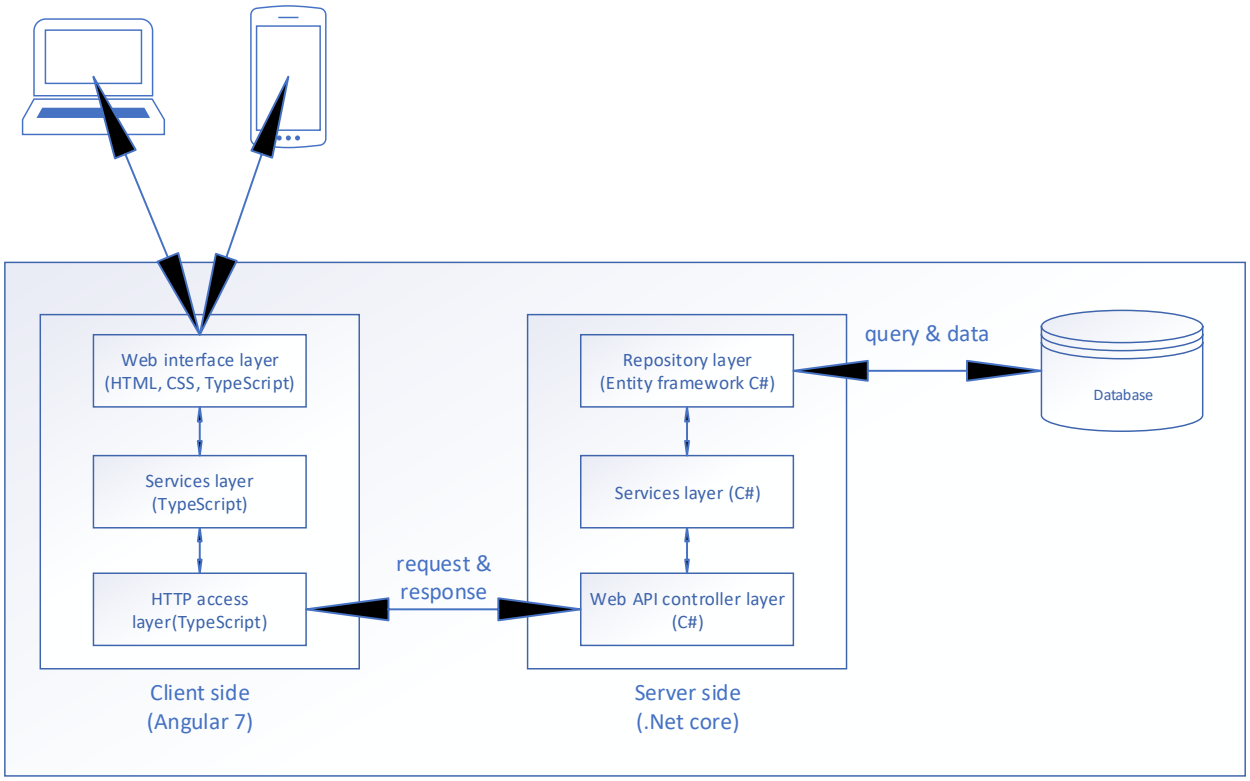


2.5.2 Data flow diagram

Data flow from client to server to database goes through a series of layers to make this application more decoupled.

- Client web interface layer has the basic HTML, CSS and TypeScript files. They are integrated as a component in Angular. This is the layer which interacts with users directly.
- Client service layer has TypeScript code contains logic that process front end data. For outgoing requests, this layer calls HTTP access layer to send requests. For incoming responses, it returns data to web interface layer to display data.
- Client HTTP access layer builds HTTP request strings to call server Web API.
- Server Web API controller layer works as a request triage. A particular HTTP request string built by client HTTP access layer will call a designated C# Web API method. For incoming requests, the controller layer will call server service layer to process the request in detail. For outgoing response, it will return data or error messages to client.
- Server service layer contains the C# logic to process HTTP requests, but it won't interact will database directly. For incoming requests, when service layer decides what data to operate, it will call server repository layer to access database. For outgoing responses, it will transform data retrieved from database to a client data view, then pass the view back to controller layer.
- Server repository layer uses Microsoft Entity Framework to select, insert, update and delete database data without knowing any detail business logic. In case when we need

to change database from SQLite to SQL Server, developers only need to modify C# code in this layer and won't affect other C# code.

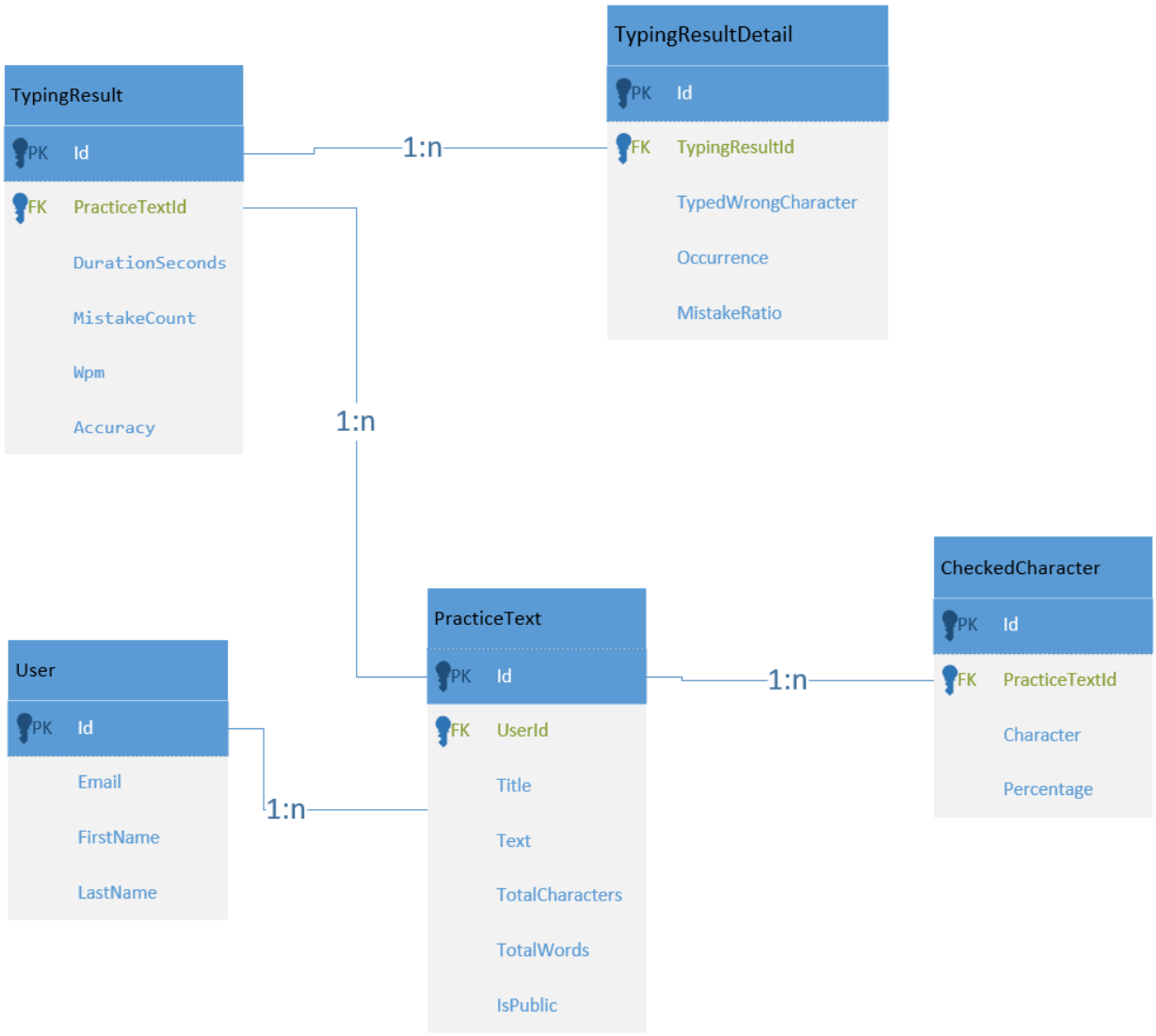


System boundry

2.5.3 Entity relationship and object diagram

All entities have a column of Globally Unique Identifier (GUID) as primary key.

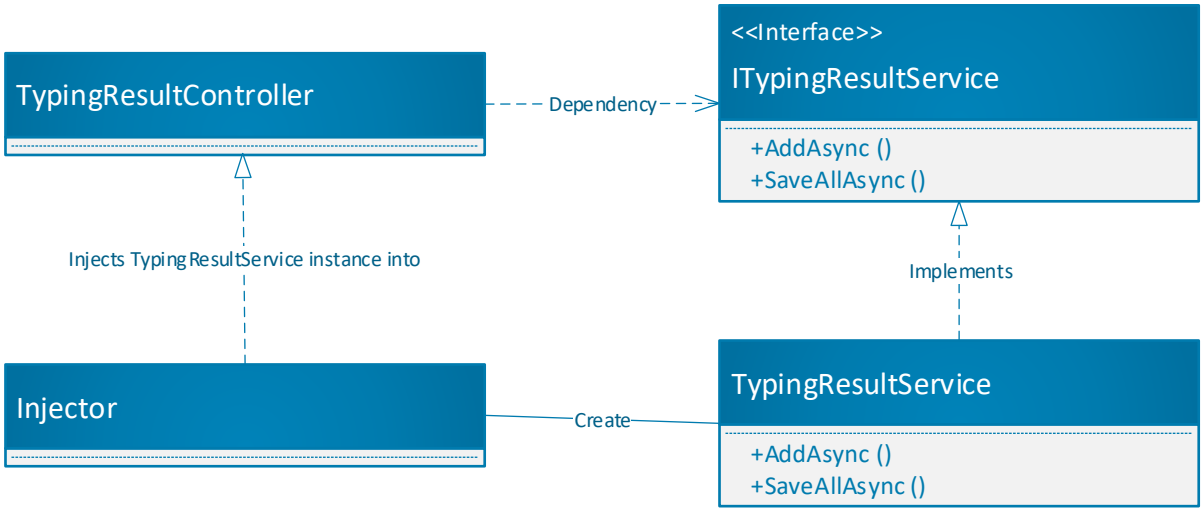
- User is the registered user who can practice typing and save practice texts. One user can save multiple practice texts into database.
- Practice Text entity has basic information about text title, number of words and characters etc. Each practice text associates with a set of checked characters. Each practice text can have multiple typing result, because it can be typed multiple times.
- Checked Characters are a set of predefined characters which will be checked for correctness in typing. If a particular checked character does not appear in a practice text, then it still has a record shows percentage 0.
- Typing Result entity has the overall information for a particular practice. Each typing result has multiple typing result detail records associate with it.
- Typing Result Detail entity has the record for a particular character typing result in a particular practice text.



2.5.4 Design pattern diagram

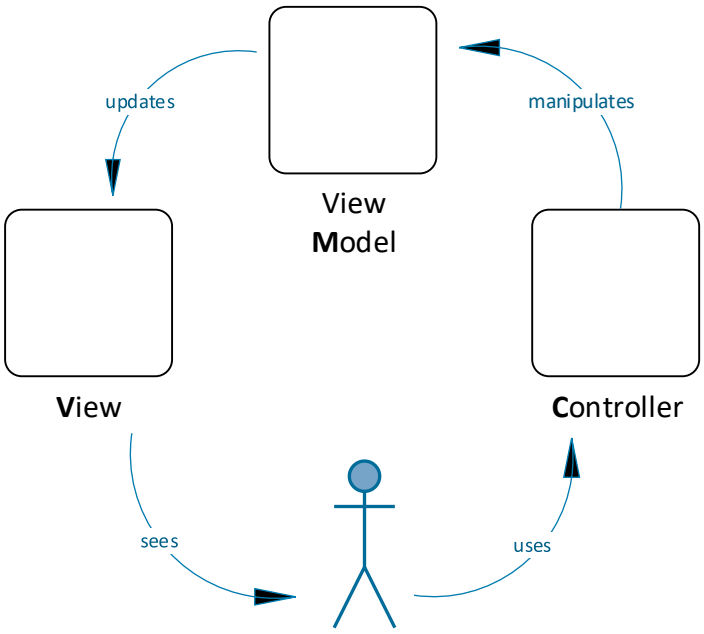
Dependency injection (DI) design pattern: All service and repository classes use dependency injection. To inject an instance instead of creating the instance by “new” keyword will make the application more decoupled, and it will be much easier for unit test.

Use “TypingResultService” class as an example for the diagram. The “TypingResultController” class needs a “TypingResultService” object, but creates such an object directly will add a new dependency that makes the application more coupled. The solution is to use the Injector provided by ASP.NET Core. We only need to register the relation between “TypingResultService” and its interface “ITypingResultService”, then the ASP.NET Core can inject a “TypingResultService” instance in the constructor of “TypingResultController” class by using interface “ITypingResultService.



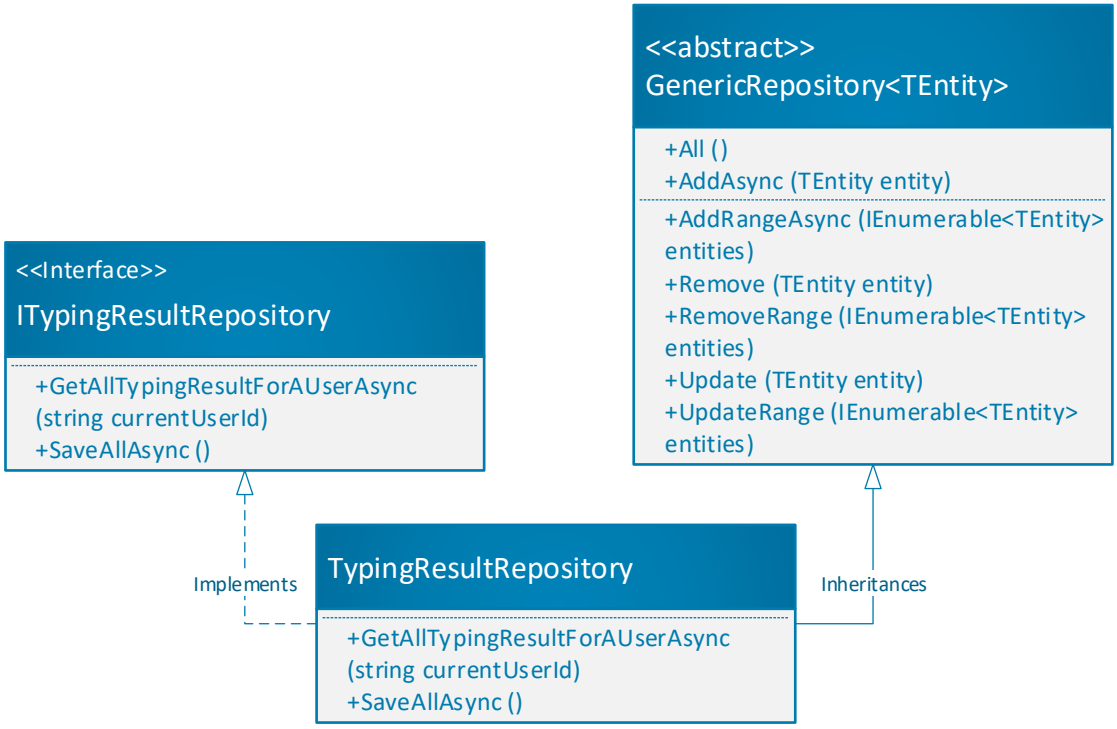
MVC (Model – View - Controller)

Angular front-end uses MVC to make the application structure clearer. The View is what users see on the web page, for example a list of practice texts the user wants to manage. The ViewModel is a TypeScript practice text object list that provides data. The data binding is dynamic, which means when the practice text data list gets changed, then what the user sees on the web page also get changed. The Controller is the logic to manipulate the data list. It hides the logic detail from user's view.

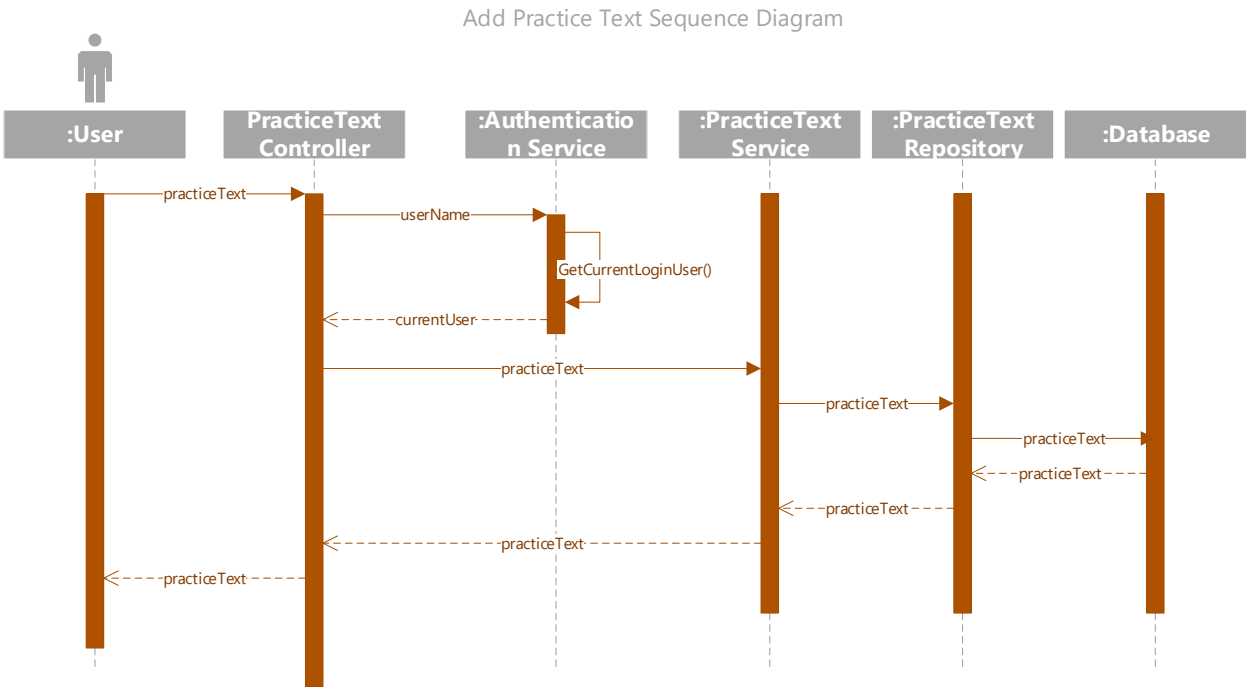


Repository design pattern

Each database table needs select, insert, update and delete operation. There are some basic database operations can be written into a generic abstract C# file, such as “GenericRepositoty” in this project. It has all the basic database operation code implemented already. A particular class “TypingResultRepositoty” only needs to inherit from the generic abstract class, then it can have all basic database operations without duplicating the code. If “TypingResultRepositoty” needs its own particular database access methods, for example “GetAllTypingResultForAUserAsync”, then it can create its own interface and implement the extra methods it needs.

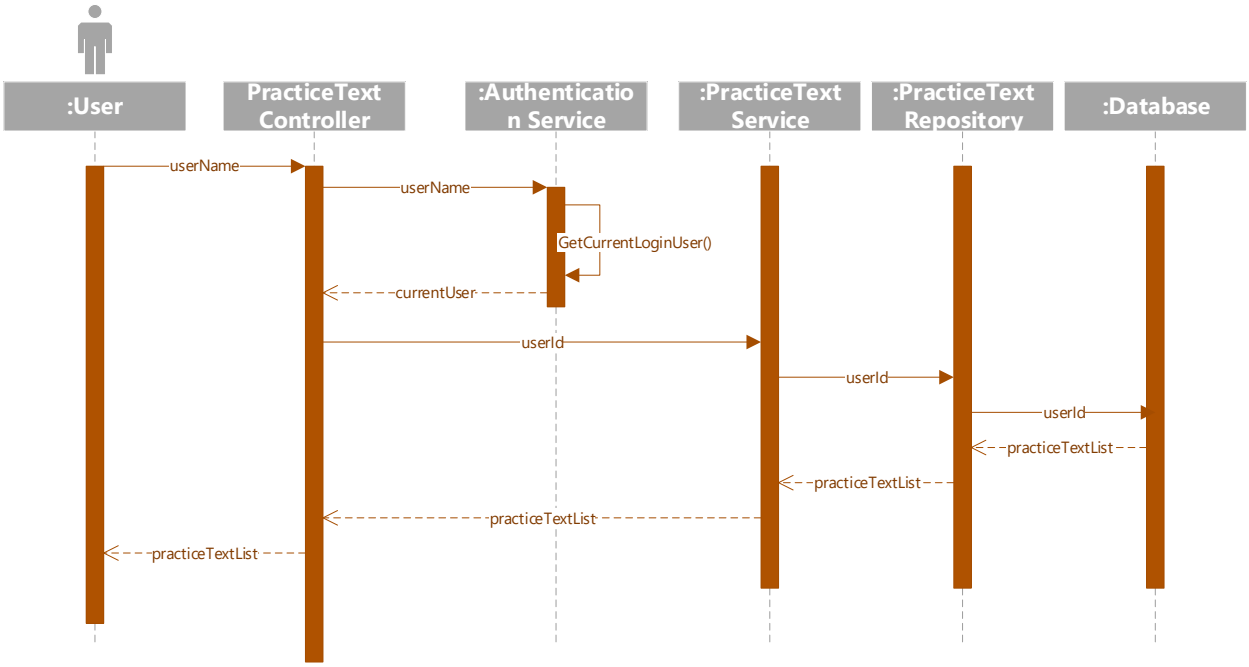


2.5.5 Sequence diagram



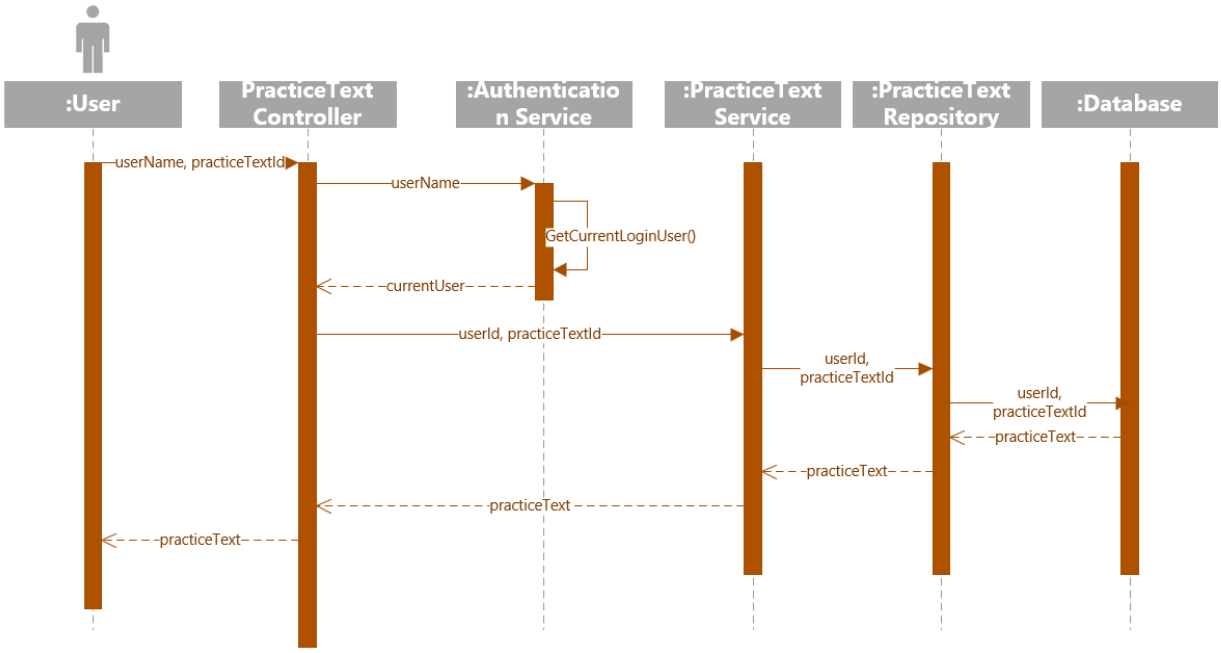
When user sends a request to add a practice text, the PracticeTextController receives new practice text data. The controller will first call Authentication service to get and verify the current login user. Then new practice text data will be sent to PracticeTextService to do the "Insert" operation. The "PracticeTextService" will create a practice text object according to the data received from user then send to PracticeTextRepository to insert this object to database. After data insertion, the code will return from the call stack all the way back to the controller. The controller will return the new practice text data to the user.

Get All Practice Text for A User Sequence Diagram



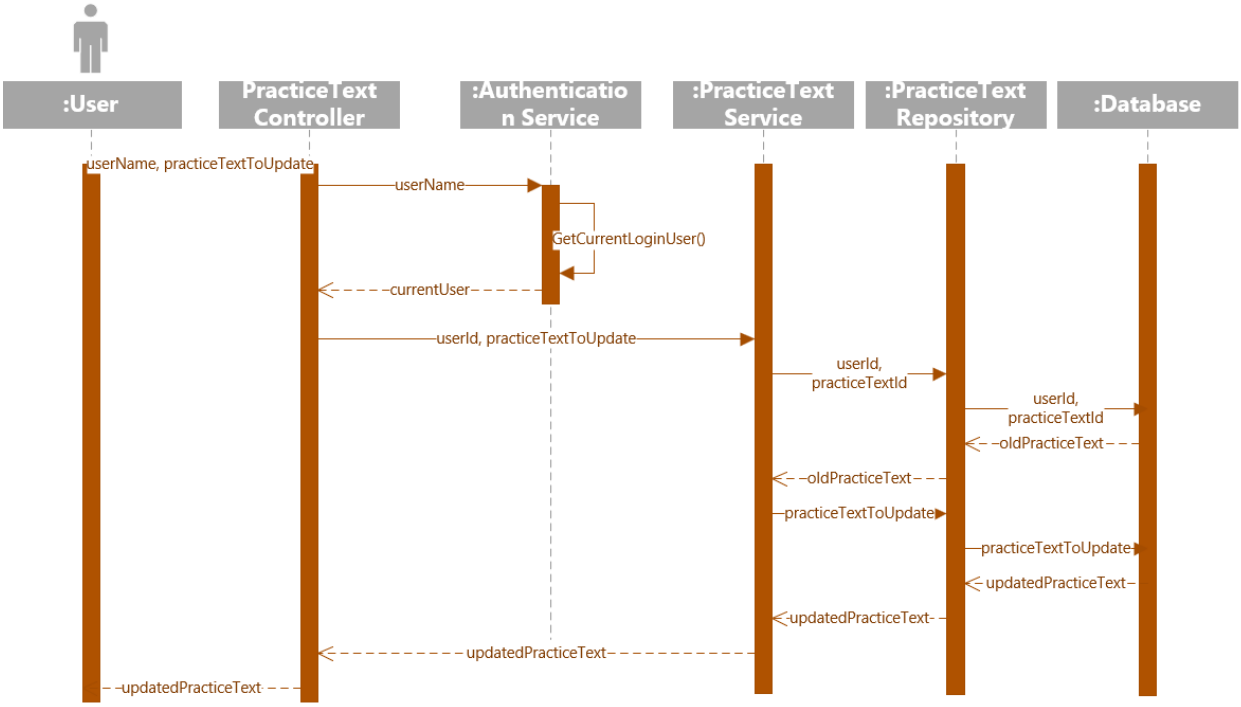
When user sends a request to get all practice texts for a user, the PracticeTextController receives a user name. The controller will first call Authentication service to get and verify the current login user. Then user Id will be sent to PracticeTextService to do the “Select” operation. The “PracticeTextService” will send the user Id to PracticeTextRepository to query the database. After PracticeTextRepository gets a list of practice text data, the code will return the list all the way back to the controller. The controller will return the practice text data list to the user.

Get A Practice Text Sequence Diagram



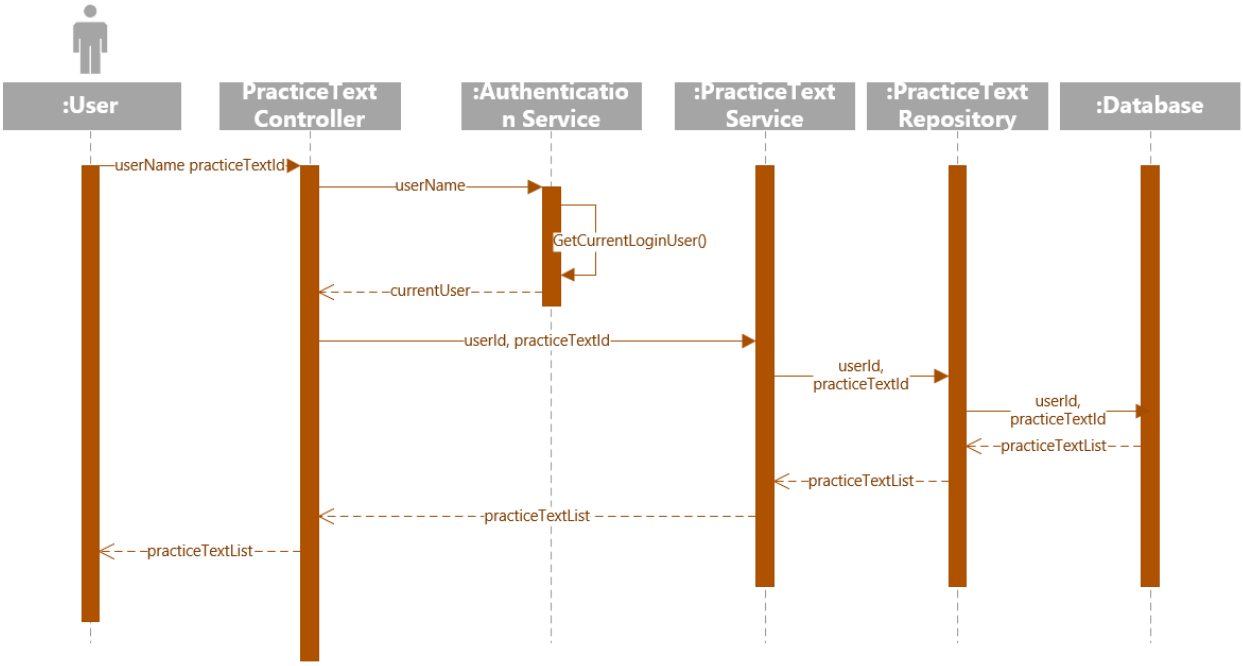
When user sends a request to get a particular practice texts for a user, the PracticeTextController receives user name and practice text Id. The controller will first call Authentication service to verify and get the current login user. Then the practice text Id and user Id will be sent to PracticeTextService to do the “Select” operation. The “PracticeTextService” will send the practice text Id and user Id to PracticeTextRepository to query the database. After PracticeTextRepository gets a practice text, the code will return the practice text all the way back to the controller. The controller will return the practice text to the user.

Update A Practice Text Sequence Diagram

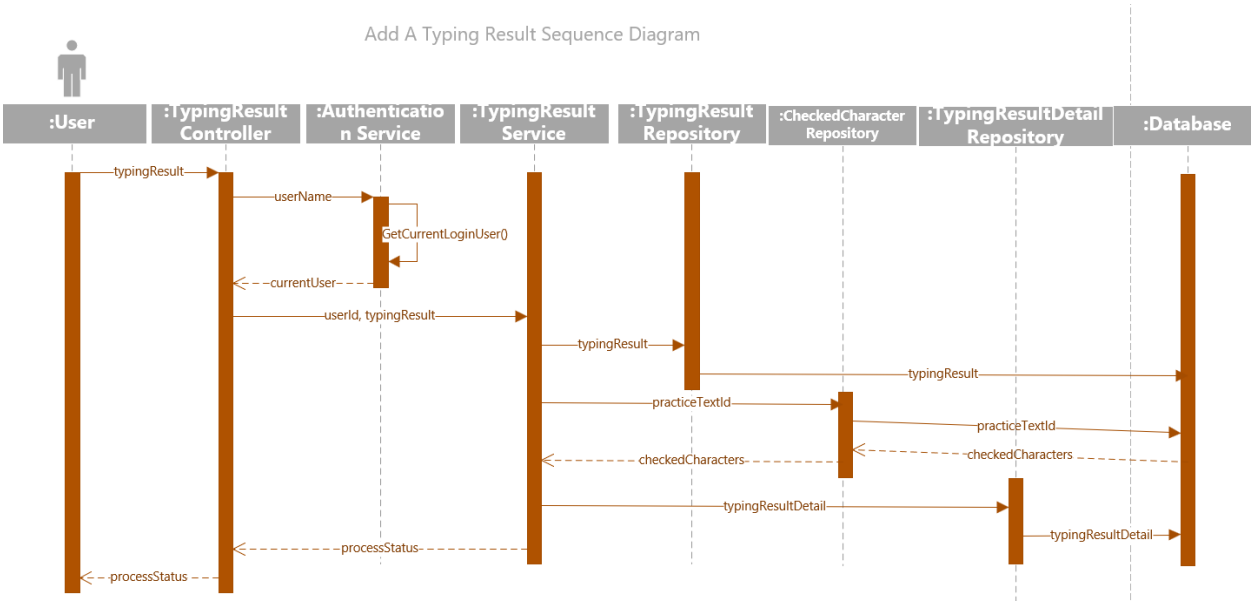


When user sends a request to update a particular practice texts for a user, the PracticeTextController receives user name and the updated practice text with existing Id. The controller will first call Authentication service to verify and get the current login user. Then the updated practice text and user Id will be sent to PracticeTextService to do the “Update” operation. The “PracticeTextService” will use the practice Id and user Id to retrieve the current practice text, then send the new practice text PracticeTextRepository to update the database. After PracticeTextRepository updates the practice text, the code will return the new updated practice text all the way back to the controller. The controller will return the practice text to the user.

Delete A Practice Text Sequence Diagram

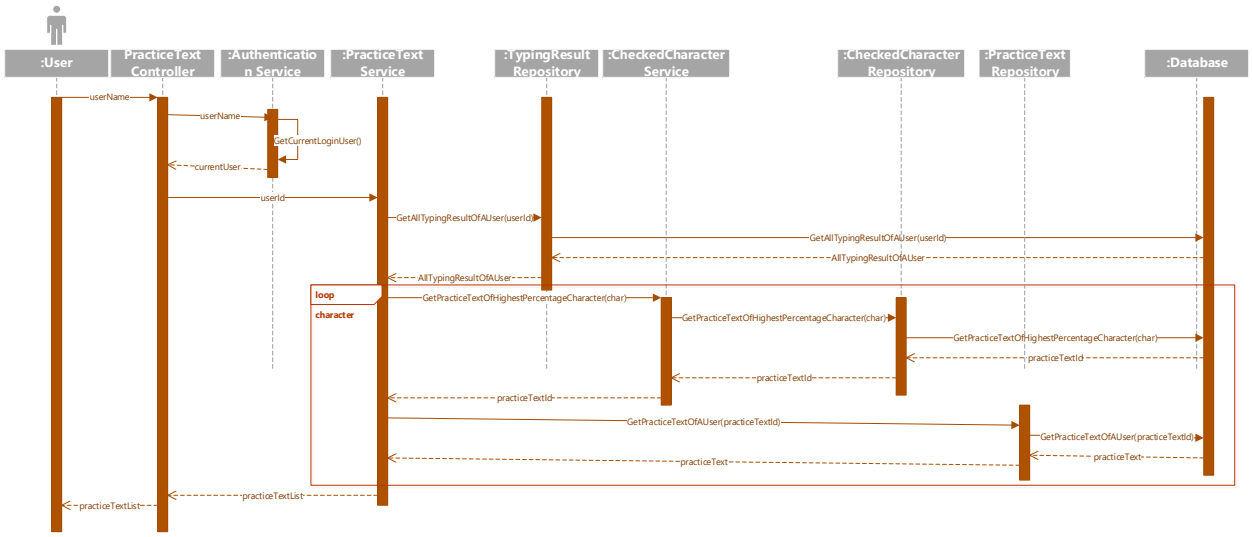


When user sends a request to delete a particular practice texts for a user, the PracticeTextController receives user name and practice text Id. The controller will first call Authentication service to verify and get the current login user. Then the practice text Id and user Id will be sent to PracticeTextService to do the “Delete” operation. The “PracticeTextService” will send the practice text Id and user Id to PracticeTextRepository to delete the record from database. After PracticeTextRepository deletes a practice text, the code will return the new practice text list all the way back to the controller. The controller will return the practice text list to the user.



When user sends a request to add a typing result of a practice texts for a user, the TypingResultController receives typing result data including user name and typing result data. The controller will first call Authentication service to verify and get the current login user. Then the typing result and user Id will be sent to TypingResultService to do the “Insert” operation. The “PracticeTextService” will send the practice text Id and user Id to TypingResultRepository to add the record to database. Then TypingResultService will send the practice text Id to CheckedCharacterRepositoty to retrieve a list of checked characters. According to the checked character list, TypingResultService will send the typing result detail to TypingResultDetailRepository to insert these data to TypingResultDetail table in database. Finally, the code will return the request status all the way back to the controller. The controller will return the status to the user.

Get Best Practice Text Sequence Diagram



When user sends a request to get a list of best practice text according to previous typing result, the PracticeTextController will get the user name. The controller will first call Authentication service to verify and get the current login user. Then the controller sends user Id to PracticeTextService. Then the user Id will be sent to TypingResultRepository to get this user's previous typing results from database. These typing result will be returned to PracticeTextService for calculation.

PracticeTextService will loop through the typing result to call CheckedCharacterService to get the highest percentage character which typed wrong in the text, then use CheckedCharacterRepository to get the practice text Id according to each character. These practice Ids will be returned to PracticeTextService.

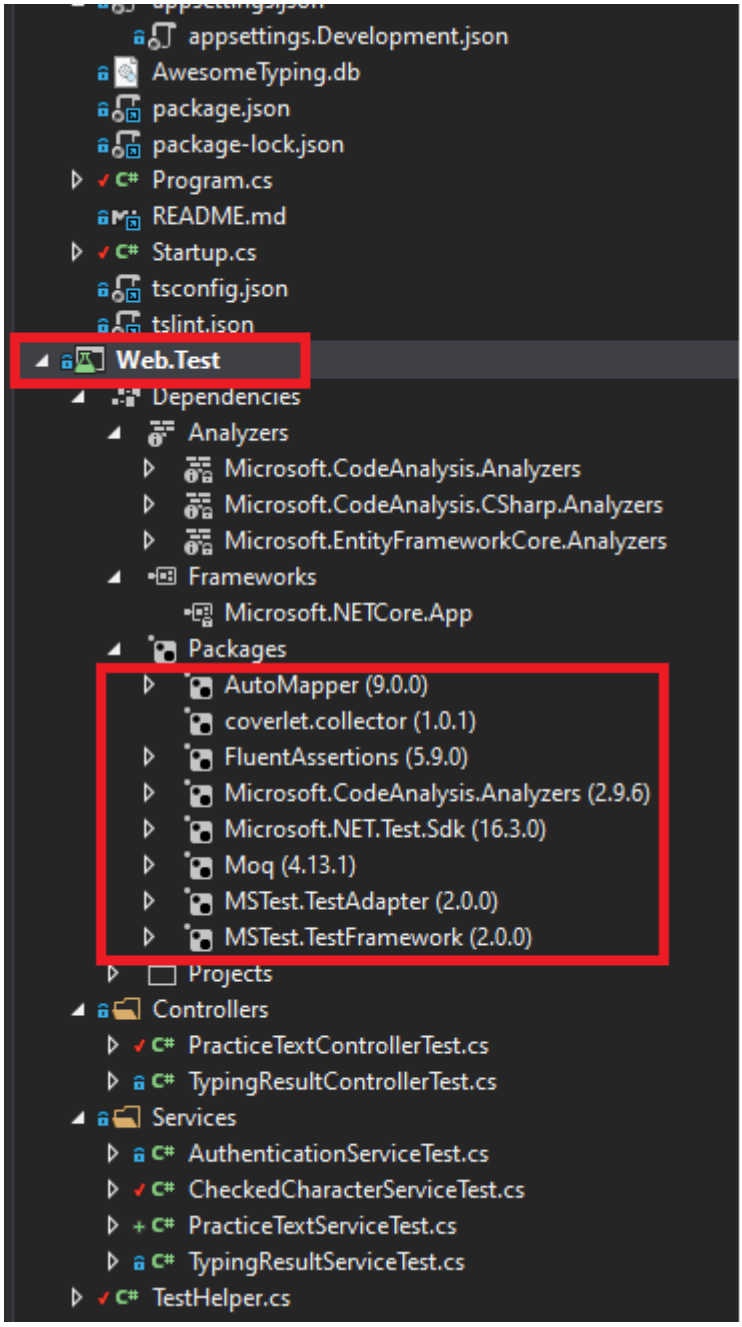
PracticeTextService will send these Ids to PracticeTextRepository to get a list of practice texts, then return the practice text list all the way back to PracticeTextController. The controller will return the best practice text list to the user.

2.6. Testing Details and Results

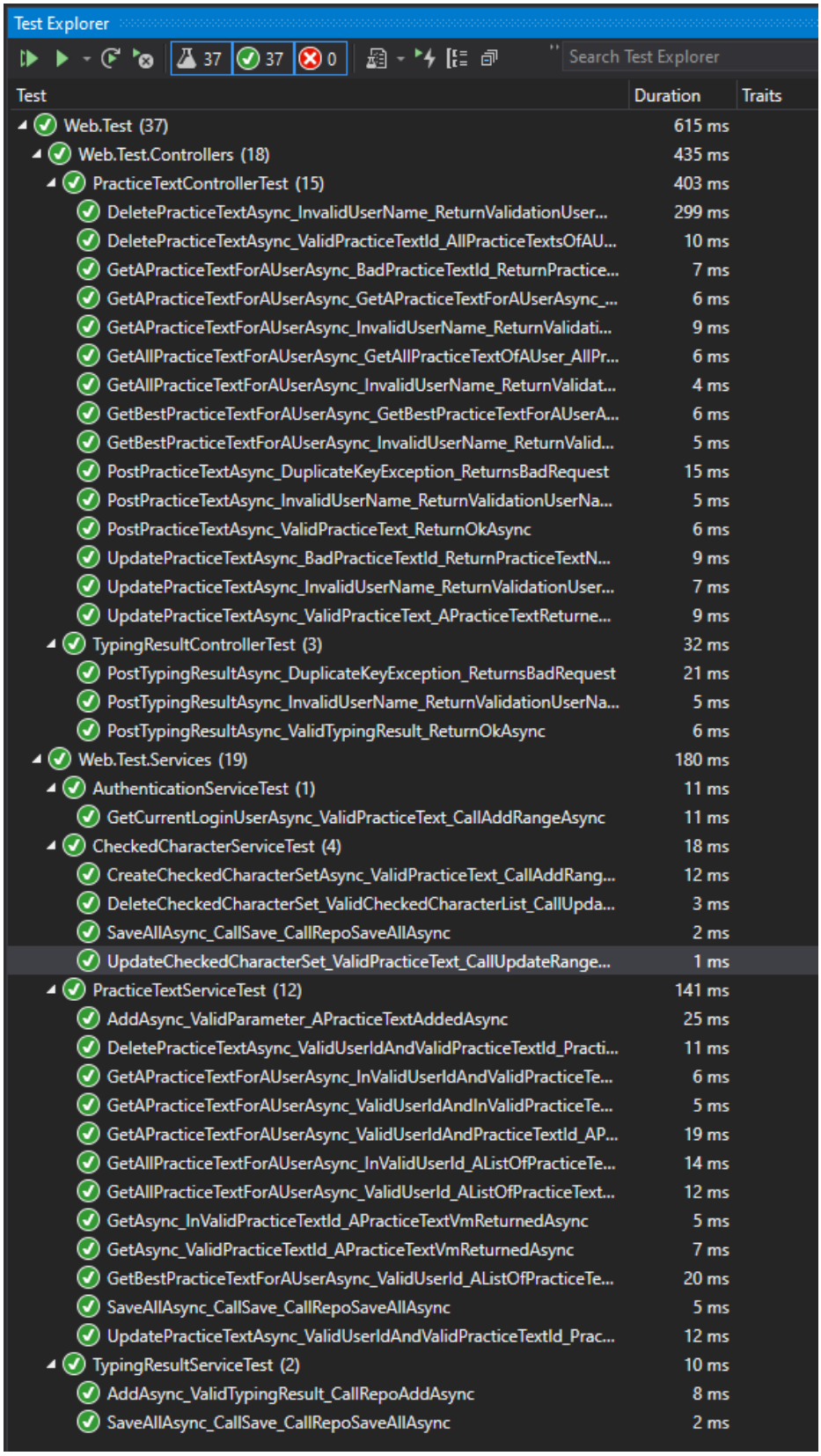
2.6.1Unit Test:

Unit test mainly uses the following tools as shown in the screenshot. In the project, the unit test module is a sub project.

- MS Test Framework
- Moq
- Fluent Assertions



The list of unit test methods is in the next screenshot followed by a description of each test.



1, DeletePracticeTextAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends an invalid user name to delete a practice text, the PracticeTextController will return a bad request status with an exception message of a valid user name required.

2, DeletePracticeTextAsync_ValidPracticeTextId_AllPracticeTextsOfAUserReturnedAsync

When user sends a valid user name to delete a practice text, the PracticeTextController will return a success HTTP status code.

3, GetAPracticeTextForAUserAsync_BadPracticeTextId_ReturnPracticeTextNotFoundAsync

When user sends a bad practice text Id to get a practice text, the PracticeTextController will return a bad request status with an exception message of not found.

4, GetAPracticeTextForAUserAsync_GetAPracticeTextForAUserAsync_APracticeTextReturnedAsync

When user sends a good practice text Id to get a practice text, the PracticeTextController will return the found practice text view model and the view model Id should be the same as practice Id passed in as parameter.

5, GetAPracticeTextForAUserAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends an invalid user name to get a practice text, the PracticeTextController will return a bad request status with an exception message of a valid user name required.

6, GetAllPracticeTextForAUserAsync_GetAllPracticeTextOfAUser_AllPracticeTextsOfAUserReturnedAsync

When user sends a good user Id to get all practice text for this user, the PracticeTextController will return a list of practice text view models with success HTTP status code.

7, GetAllPracticeTextForAUserAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends an invalid user name to get all practice text for this user, the PracticeTextController will return a bad request status with an exception message of a valid user name required.

8, GetBestPracticeTextForAUserAsync_GetBestPracticeTextForAUserAsync_BestPracticeTextsReturnedAsync

When user sends a good user Id to get best practice text for this user, the PracticeTextController will return a list of best practice text view model with success HTTP status code.

9, GetBestPracticeTextForAUserAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends an invalid user Id to get best practice text for this user, the PracticeTextController will return a bad request status with an exception message of a valid user name required.

10, PostPracticeTextAsync_DuplicateKeyException_ReturnsBadRequest

When user sends a new practice to add with an existing practice text Id, the PracticeTextController will return a bad request status with an exception message of duplicate key.

11, PostPracticeTextAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends a new practice to add with an invalid user Id, the PracticeTextController will return a bad request status with exception message of a valid user name required.

12, PostPracticeTextAsync_ValidPracticeText_ReturnOkAsync

When user sends a new practice to add and all data are valid, the PracticeTextController will return the new added practice text view mode with a success status.

13, UpdatePracticeTextAsync_BadPracticeTextId_ReturnPracticeTextNotFoundAsync

When user sends a new practice to update and the practice text id is invalid, the PracticeTextController will return a practice text not found status.

14, UpdatePracticeTextAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends a new practice to update with an invalid user Id, the PracticeTextController will return a bad request status with exception message of a valid user name required.

15, UpdatePracticeTextAsync_ValidPracticeText_APracticeTextReturnedAsync

When user sends a new practice to update and all data are valid, the PracticeTextController will return the new updated practice text view mode with a success status, and the returned practice text Id is the same as the passed in parameter practice text Id.

16, PostTypingResultAsync_DuplicateKeyException_ReturnsBadRequest

When user sends a new typing result to add with an existing typing result Id, the TypingResultController will return a bad request status with an exception message of duplicate key.

17, PostTypingResultAsync_InvalidUserName_ReturnValidationUserNameRequiredAsync

When user sends a new typing result to add with an invalid user name, the TypingResultController will return a bad request status with an exception message of valid user name required.

18, PostTypingResultAsync_ValidTypingResult_ReturnOkAsync

When user sends a new typing result to add and all data are valid, the TypingResultController will return a success status.

19, GetCurrentLoginUserAsync_ValidPracticeText_CallAddRangeAsync

When user sends a valid user name to get the current log in user, the AuthenticationService will return the current user Id.

20, CreateCheckedCharacterSetAsync_ValidPracticeText_CallAddRangeAsync

When a valid practice text has been added to the database, the practice text's related checked characters should also be added to the database and only be added once.

21, DeleteCheckedCharacterSet_ValidCheckedCharacterList_CallUpdateRangeAsync

When a valid practice text has been deleted from the database, the practice text's related checked characters should also be deleted to the database and only be deleted once.

22, SaveAllAsync_CallSave_CallRepoSaveAllAsync

After a set of checked characters have been added, updated or deleted in the database, the checked characters table should only be committed once.

23, UpdateCheckedCharacterSet_ValidPracticeText_CallUpdateRangeAsync

When a valid practice text has been updated in the database, the practice text's related checked characters should also be updated in the database and only be updated once.

24, AddAsync_ValidParameter_APracticeTextAddedAsync

When a valid practice text needed to be added to the database, the practice text should be added to the database and only be added once.

25, DeletePracticeTextAsync_ValidUserIdAndValidPracticeTextId_PracticeTextDeletedAsync

When a valid practice text needed to be deleted from the database, the practice text should be deleted from the database and only be deleted once.

26, GetAPracticeTextForAUserAsync_InValidUserIdAndValidPracticeTextId_APracticeTextVmReturnedAsync

When the PracticeTextService needs to get a practice text with an invalid practice text id and a valid user Id, then the service should return null.

27, GetAPracticeTextForAUserAsync_ValidUserIdAndInValidPracticeTextId_APracticeTextVmReturnedAsync

When the PracticeTextService needs to get a practice text with a valid practice text id and an invalid user Id, then the service should return null.

28, GetAPracticeTextForAUserAsync_ValidUserIdAndPracticeTextId_APracticeTextVmReturnedAsync

When the PracticeTextService needs to get a practice text with a valid practice text id and a valid user Id, then the service should return the practice text view model and the view model Id is the same as the practice Id passed in as parameter.

29, GetAllPracticeTextForAUserAsync_InValidUserId_AListOfPracticeTextVmReturnedAsync

When the PracticeTextService needs to get all practice texts for a user, and the user Id is invalid, then the service should return an empty list.

30, GetAllPracticeTextForAUserAsync_ValidUserId_AListOfPracticeTextVmReturnedAsync

When the PracticeTextService needs to get all practice texts for a user, and the user Id is valid, then the service should return a list of practice text view models.

31, GetAsync_InValidPracticeTextId_APracticeTextVmReturnedAsync

When the PracticeTextService needs to get a practice texts according to a practice Id, and the Id is invalid, then the service should return null.

32, GetAsync_ValidPracticeTextId_APracticeTextVmReturnedAsync

When the PracticeTextService needs to get a practice texts according to a practice Id, and the Id is valid, then the service should return a practice text view model, and the model's Id should be the same as the passed in parameter Id.

33, GetBestPracticeTextForAUserAsync_ValidUserId_AListOfPracticeTextVmReturnedAsync

When the PracticeTextService needs to get a list of best practice texts according for a user, and the user Id is valid, then the service should return a list of practice text view models.

34, SaveAllAsync_CallSave_CallRepoSaveAllAsync

After a practice text has been added, updated or deleted in the database, the practice text table will be committed once.

35, UpdatePracticeTextAsync_ValidUserIdAndValidPracticeTextId_PracticeTextUpdatedAsync

When the PracticeTextService needs to update a practice text with a valid practice text id and a valid user Id, then the service should only update the model once.

36, AddAsync_ValidTypingResult_CallRepoAddAsync

When the TypingResultService needs to add a valid typing result, the TypingResultRepositopty should only add typing result once.

37, SaveAllAsync_CallSave_CallRepoSaveAllAsync

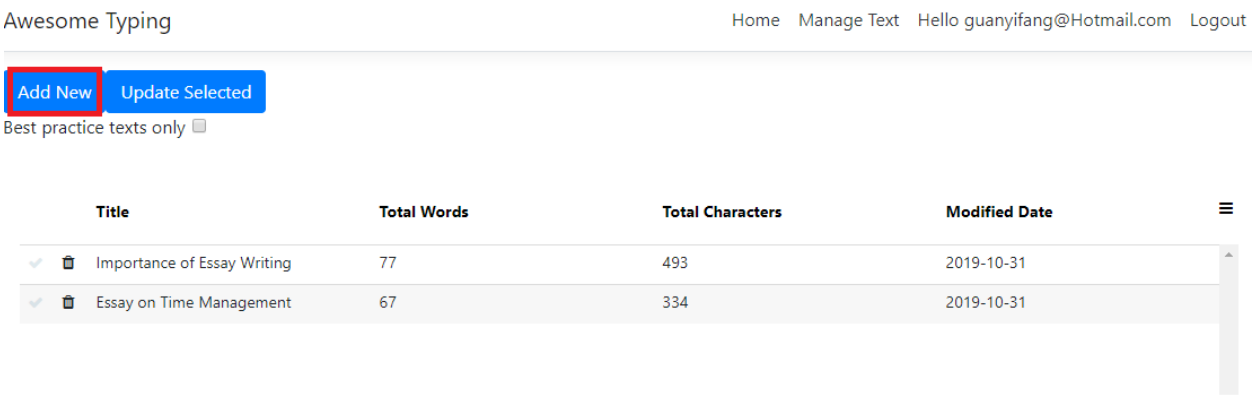
After a typing result record has been added, updated or deleted in the database, the typing result table will be committed once.

2.6.2 Feature Manual Test

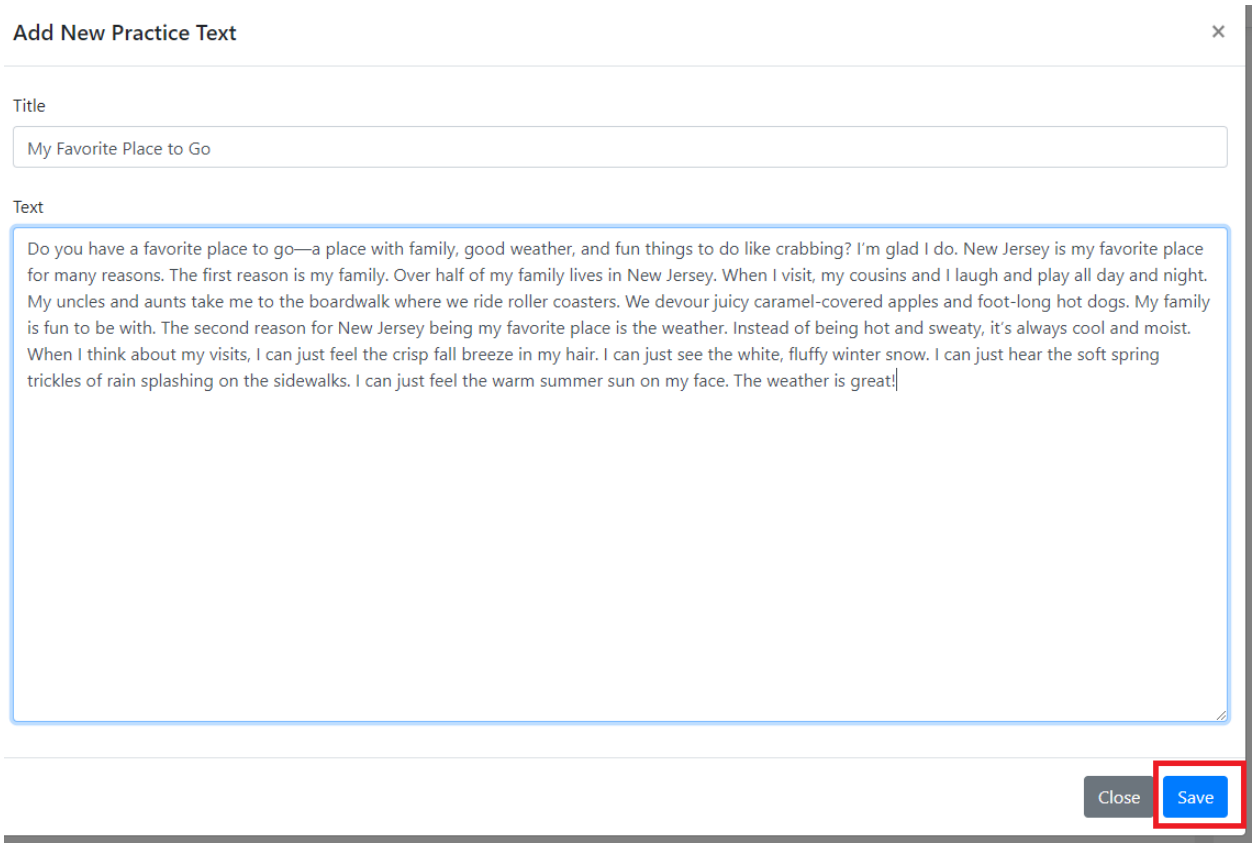
This test is to test the entire data flow of a use case and to make sure the web interface shows what it expects.

Add Practice Text

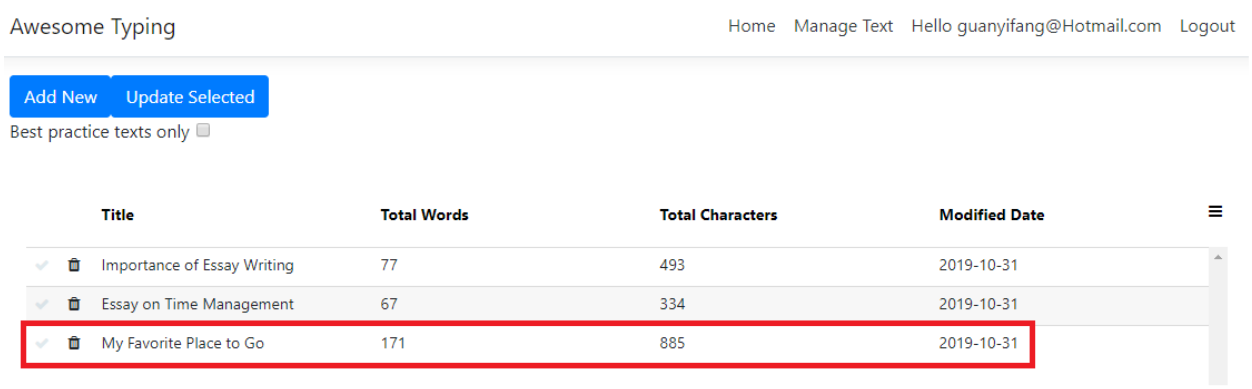
1, User clicks the “Add New” button to see a pop-up form with text edit field.



2, User can add and edit practice text then click Save.

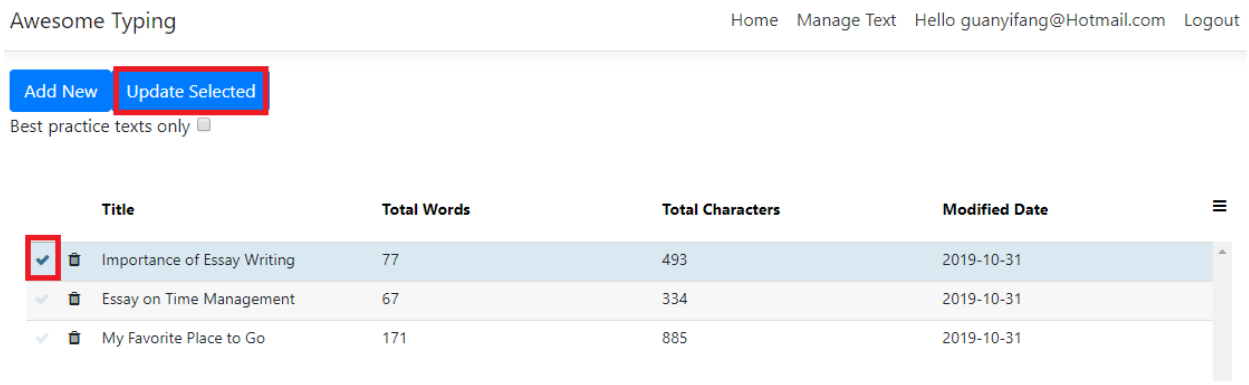


3, The form is closed automatically then back to the practice text list again. User can see the updated practice text list with the new added practice text.

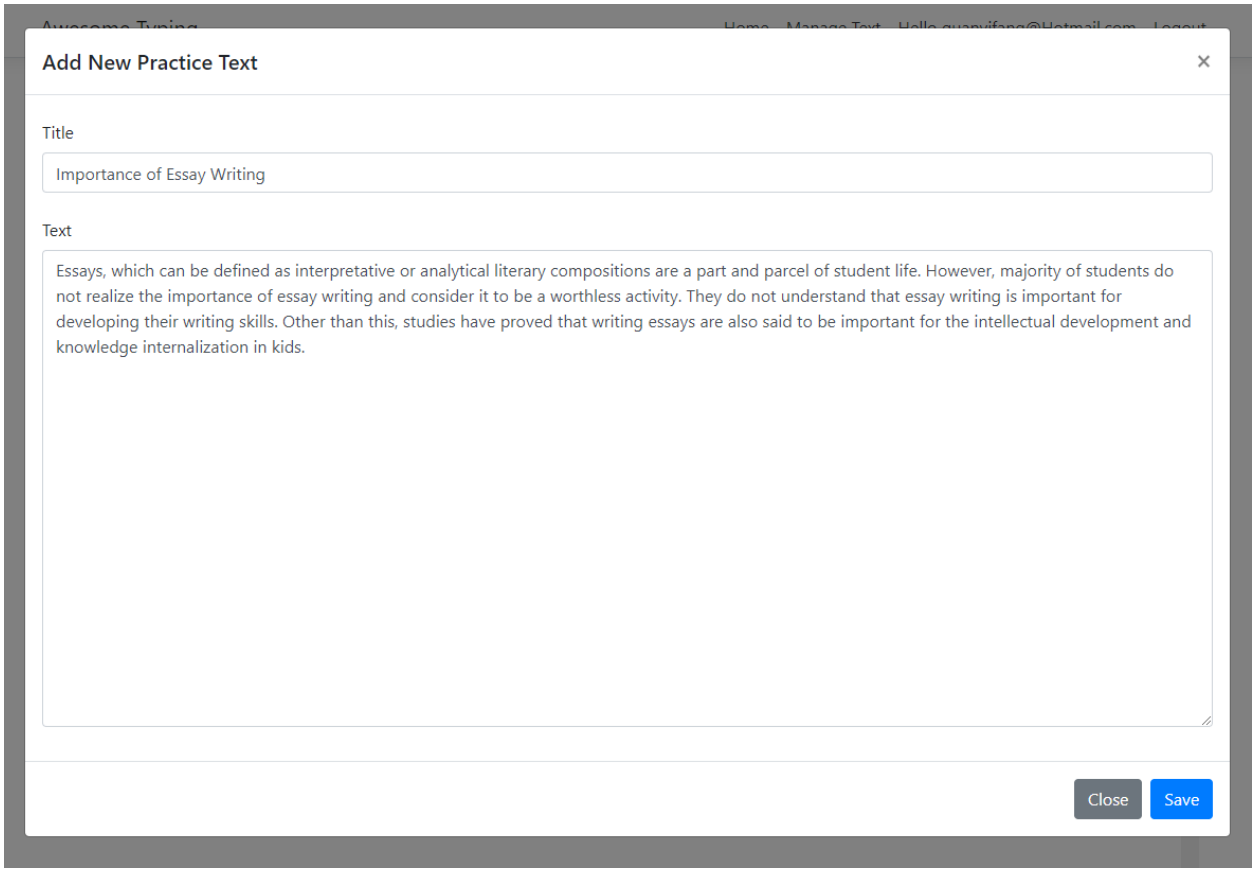


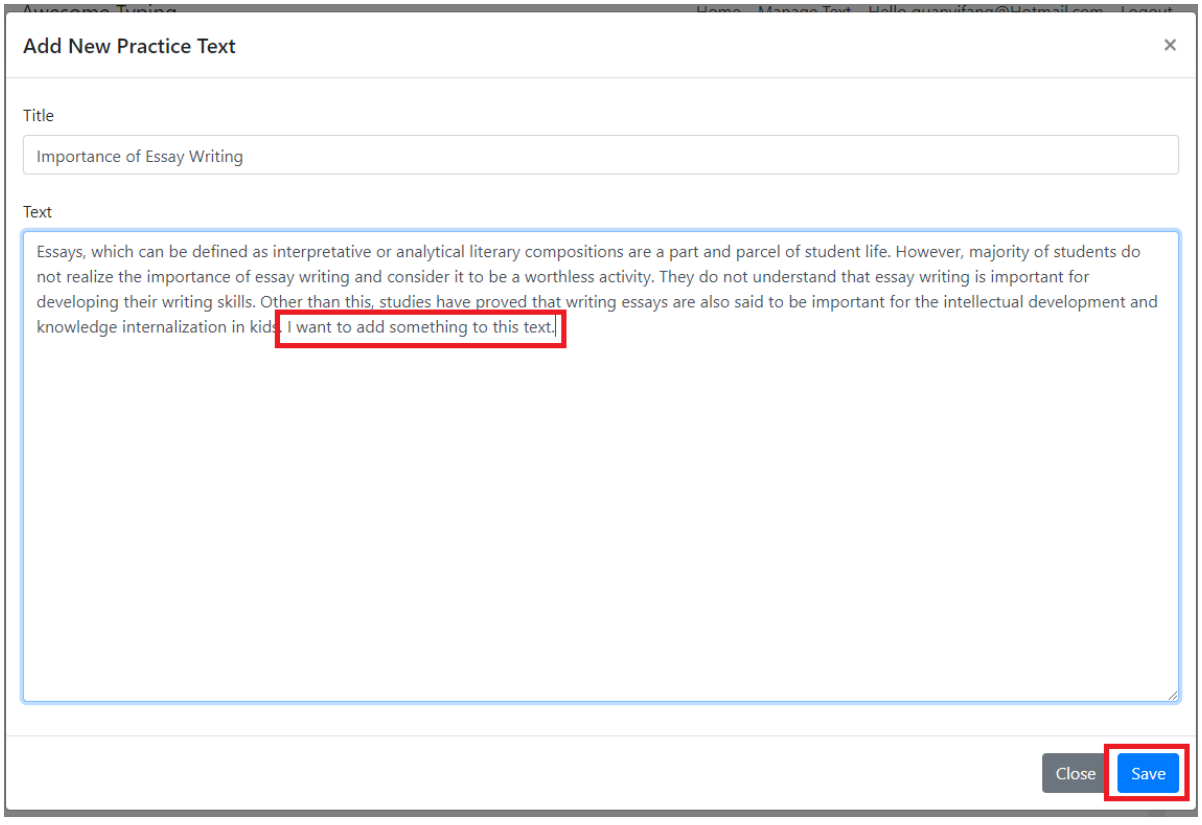
Update Practice Text

1, User can select a practice text in the list, then click the “Update Selected” button to see a pop-up form with current practice text loaded.

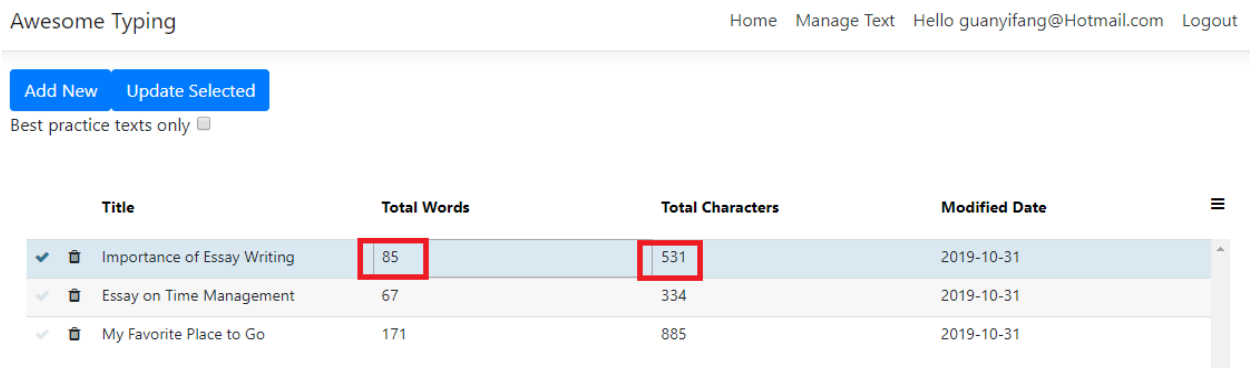


2, User can edit the practice text and click Save.


Add New Practice Text
Title
Importance of Essay Writing
Text
Essays, which can be defined as interpretative or analytical literary compositions are a part and parcel of student life. However, majority of students do not realize the importance of essay writing and consider it to be a worthless activity. They do not understand that essay writing is important for developing their writing skills. Other than this, studies have proved that writing essays are also said to be important for the intellectual development and knowledge internalization in kids.
Close
Save

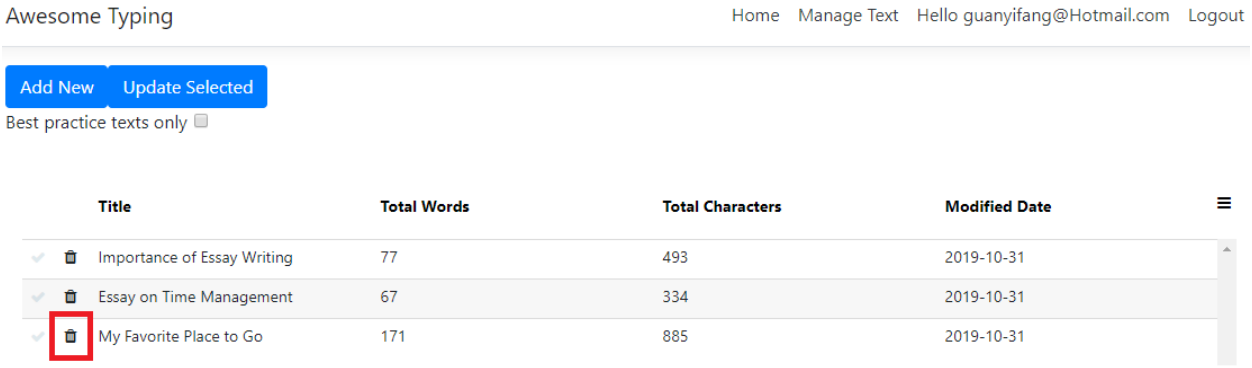

Add New Practice Text
Title
Importance of Essay Writing
Text
Essays, which can be defined as interpretative or analytical literary compositions are a part and parcel of student life. However, majority of students do not realize the importance of essay writing and consider it to be a worthless activity. They do not understand that essay writing is important for developing their writing skills. Other than this, studies have proved that writing essays are also said to be important for the intellectual development and knowledge internalization in kids. I want to add something to this text.
Close
Save

3, User will see the selected practice data fields is updated.

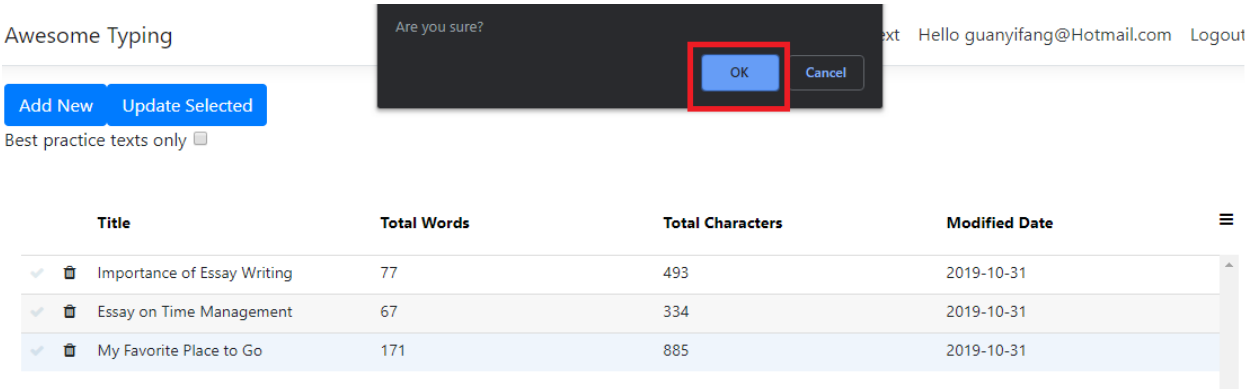


Delete Practice Text

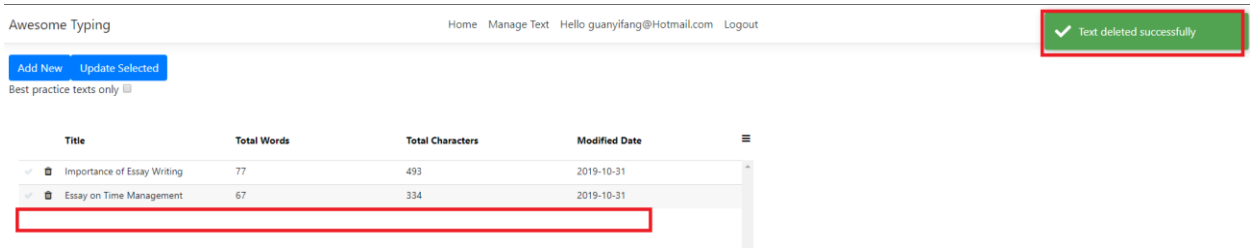
1, User can click the recycle bin button in front of the practice text record to delete it.



2, User clicks OK button to confirm delete.

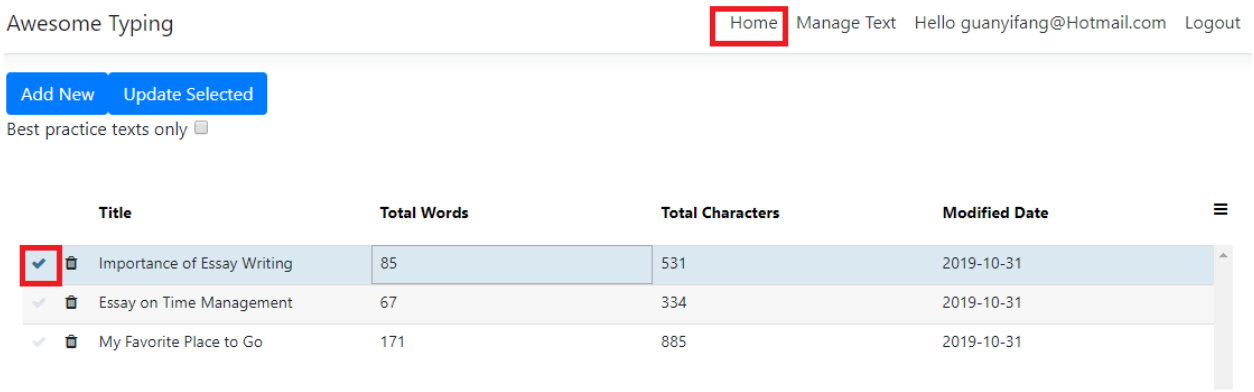


3, Then user will no longer see the deleted practice text.

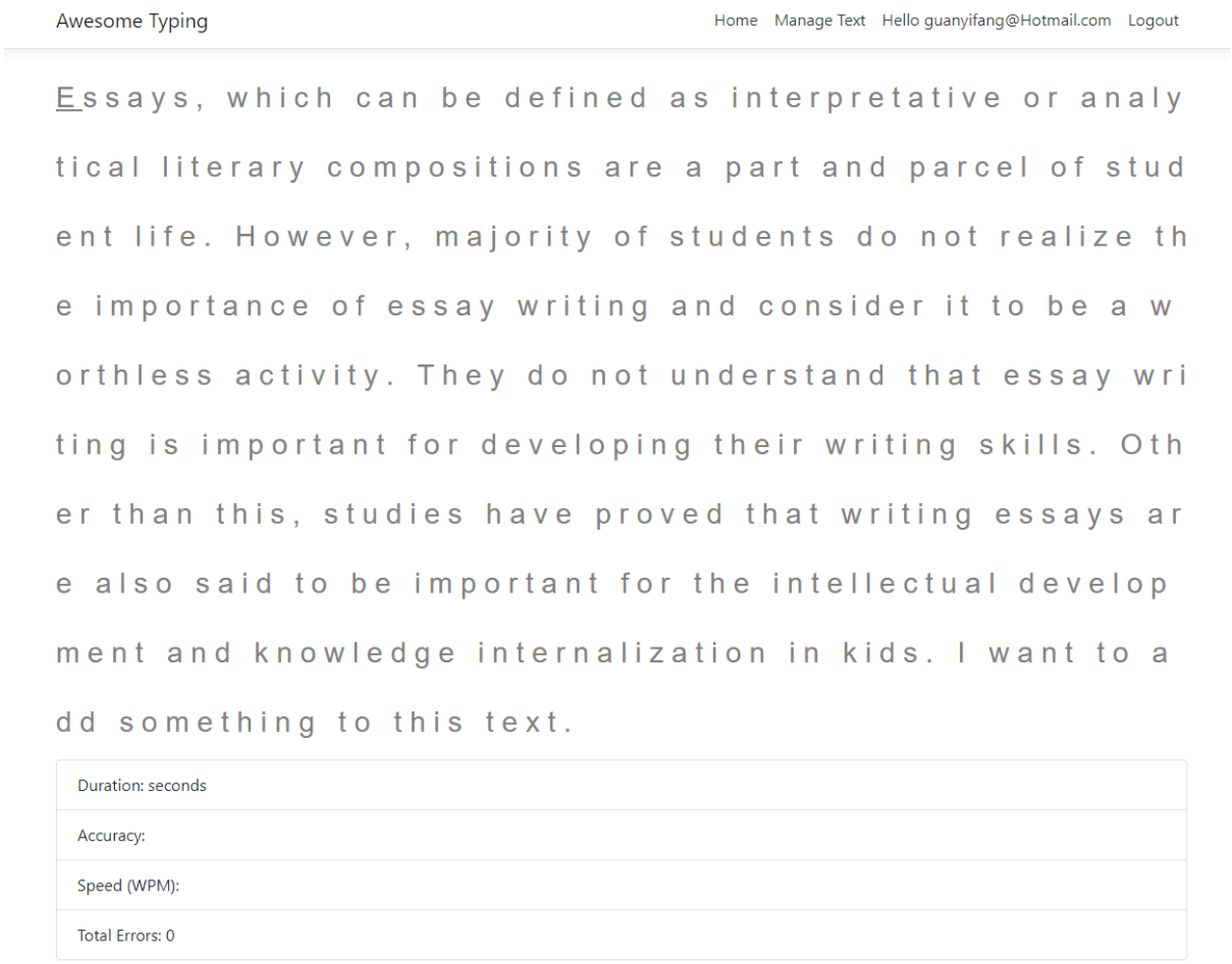


Select a Practice Text

1, When the user wants to select a practice text to type, user can click the record line to highlight it. The check mark in front indicates this is the current selected practice text to type. Then user clicks the Home button to see the loaded text to type.

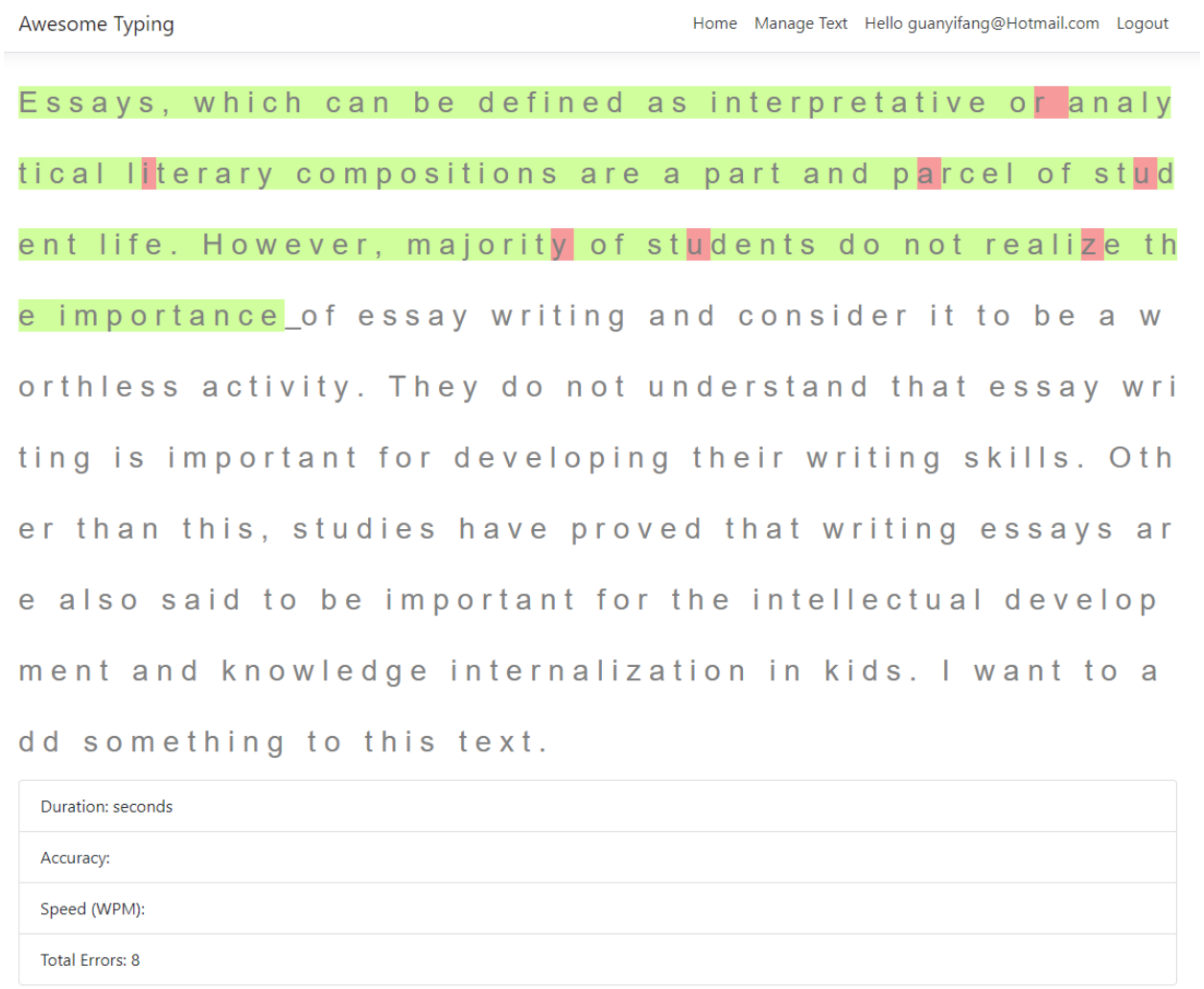


2, Before typing, the loaded text background are white. The background timer will not start until the first key pressed.



Typing

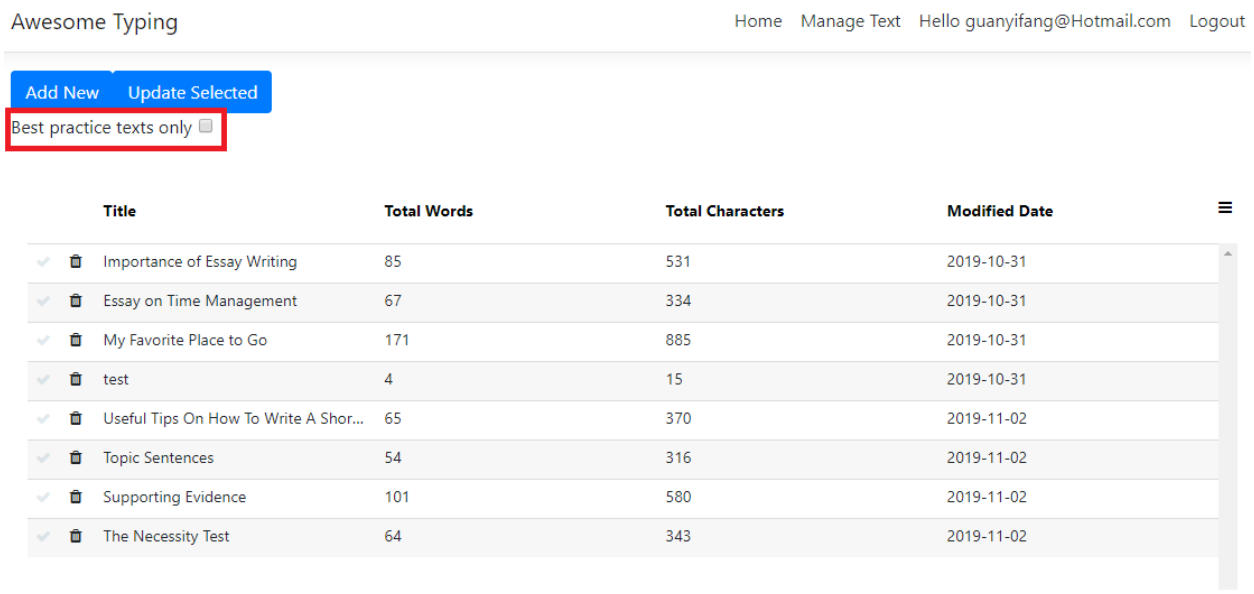
1, When user starts typing, the background of the correct keys typed will turn to green, otherwise it will turn to red.

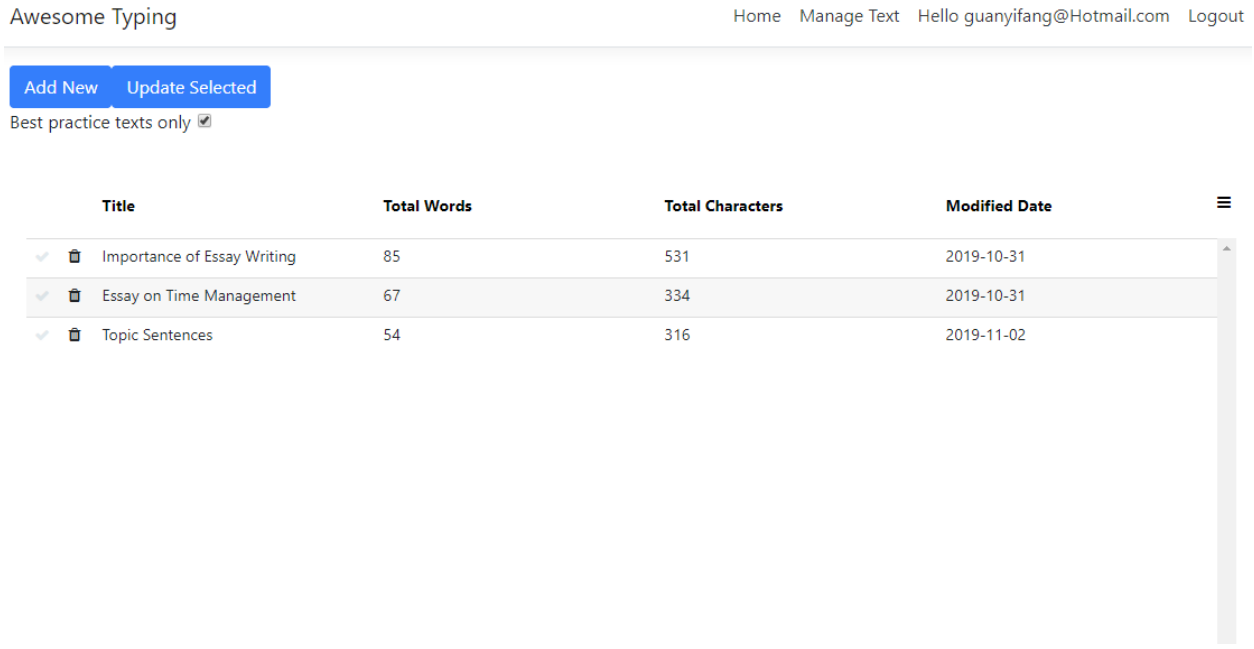


2, When typing is finished, the typing statistics will appear and the result will be sent to web server to save in database.

Get Best Practice Text

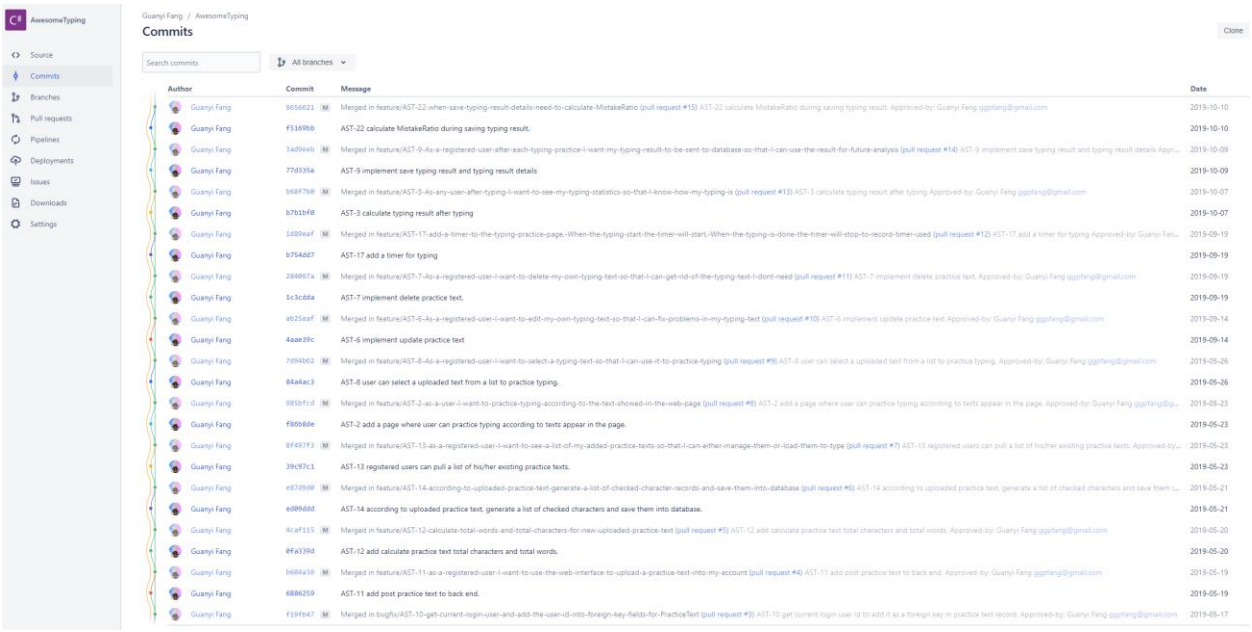
If the user only wants to select the best practice test according to previous typing results, the user will check the “Best practice texts only” box, then only the best practice texts will show up. Uncheck this box will show all saved practice texts of this user.





2.7. Implications of Implementation

I used Agile scrum methodology to manage the process of this project. I break this project into many small issues and I commit code according to each issue into Bitbucket. The screenshot shows on page of all my commits as an example.



I implement this application in three parts: front end, back end and database.

Front end: I use Angular 7 and Bootstrap 4 to implement the web site. The web site has 3 Angular components which include web pages, style sheets, and TypeScript files. I put all the data models in a separate folder. I also put global shared data such as “default empty id” which is “00000000-0000-0000-000000000001” in the data model folder. All the service Typescript files are in the service folder, for example Web API client that send data to web server.

Back end: I use Microsoft .Net Core and Entity Framework Core to implement the backend. First, I put Web API controllers in a controller folder. Controllers do not have detailed logic in them. Controllers work like triages which send income data to designated services and send data result or error messages back to client. Second, I put service logic in service folder. All services have their interfaces, so that I do not have to instantiate any service objects in any controller classes. I use “dependency injection” (DI) to instantiate these services. It will be easy for decoupling C# classes and unit test. Services will not communicate with database directly. Service will call repository classes to access database to do insert, select, update or delete (CRUD). Third, all repository classes also have their interfaces for dependency injection. Repository classes are the database direct access layer in the data process flow. This “Controller-Service-Repository” pattern can separate functions in data process flow and will be easy for future modification and enhancement. For example, if in the future I decide to switch to SQL Server instead of SQLite, I do not need to rewriter too much code. I only need to modify some code in the Repository layer. All code in Controller and Service are totally reusable.

Database: I choose SQLite as the application database because it's free and I have learned in my BCIT Android courses. SQLite has enough functionality according to the requirement of this project. I used SQLite Studio to access SQLite database directly. In the implementation of this project, I mainly used code first approach to create and update the SQLite database. Entity Framework Core (EF Core) supports code first migration which can create, modify database through C# code. Each time I run EF Core migration; EF Core will record the database scheme update in a table in SQLite database.

2.8. Innovation

To make learning more efficient, an important method is to minimize the time we practice on something we are already been good at and maximize the practice time on what we are not good at. Regrettably, I searched many typing training software for my daughter, but there is no such a typing software can act like a smart typing trainer. My online typing training application will have the innovation to fill this gap.

From technology perspective, the application I am going to develop is far from the boundary of science. The innovation selling point of my software is that currently there is no such a typing skill training software which can act like a personal typing trainer to guide each individual user through the most efficient learning path by analyzing users' typing mistakes and searching the best available typing practice texts for them. The innovation concept is not only for learning typing. I believe it is for learning anything, for example math, physics or any other topics. I believe that teachers give all students the same homework to practice is not always efficient. Each student should get different practice questions at some point of learning. If there is a software can collect all students' mistakes in their homework and tests, then analyze the root causes of why they made these mistakes and finally provide each student different practices which target their weak points during learning, it will be a big innovation for our society. I do not have the knowledge and ability to develop such a complex software, but I can use the same teaching and learning rational to start from a smaller task, an online typing application. It is not a simple improvement such as making a typing software interface more attractive or something like that. My project will give the typing software an essential innovation, that is, let a typing software have the analytical ability by collecting performance data of each user and the ability to suggest users with the exercises that best enhance their learning outcomes.

Record analysis algorithm: This algorithm can select the best saved practice text and suggest the user to practice typing according to this user's previous type errors. It will target where the most mistakes are.

1, This typing training application only check the following characters (called **checked characters**):

- Alphabet case insensitive (26 characters)
- Number keys from 0-9 (10 characters)
- Punctuations: comma, period, single, double quote, semi colon, question mark, exclamation mark (7 characters).

Totally there are 43 keys will be checked. I only use these 43 keys as an example to explain the algorithm. In the application, I can implement a user configuration that sets which characters users want to include, such as add "<" or "%" or remove number keys etc.

2, Each time when a user uploads a new essay to the typing training application, the essay will be assigned a unique id and saved into a database table called “PracticeText”. As the application checks 43 characters, a set of 43 records according to each checked character will be save to another table called “CheckedCharacter”. Each record in “CheckedCharacter” table will have a foreign key reference to “Id” column in “PracticeText” table.

The “CheckedCharacter” table has the following columns:

- Id
- PracticeTextId (Foreign Key)
- Character (‘a’ or ‘b’ or other checked characters)
- **Percentage (P)** (percentage of this check character appears in this practice text, for example 1.825) (If the user updates the practice text, this P will be calculated again)

The PracticeText table has the following columns

- Id
- UserId (Foreign Key to indicate the owner of this practice text)
- Text
- TotalCharacter (how many checked characters in this text)

Each record in “PracticeText” will have 43 associated records in “CheckedCharacter” table.

Note: in the real implementation, these tables may have more columns than showed above. Here I only show the columns relate to the algorithm)

3, After a user finishes typing a practice text, a result record will be saved into database table called “TypingResult”.

The “TypingResult” table has the following columns

- Id
- UserId (Foreign Key to indicate who typed it)
- PracticeTextId (Foreign Key)
- PracticeTime (include year, month day and time)

Note: uses can decide if the practice this time will be save or not. If they select to save the result, then it is like a typing proficiency test for them. The result data will be used for analysis in this algorithm. If they do not want to save the result, then it means they just want to do a regular practice, not a test for future analysis.

4, Then a set of 43 detailed typing records will be saved to a table called “TypingResultDetail”

The “TypingResultDetail” table has the following columns

- Id
- TypingResultId (Foreign Key to indicate which typing result it references to)
- CharacterTypedWrong (‘a’ or ‘b’ or other checked characters)

- **Occurrence (O)** (an integer which shows how many times this checked character was typed wrong)
- **MistakeRatio (MR)** ("Occurrence" divided by "Percentage" field in "CheckedCharacter" table. Larger MR indicates user is more like need more practice on this checked character)

We need "MistakeRatio" later to calculate which checked characters this user does not type well. We use "MistakeRatio", not "Occurrence", because in a longer practice text, user have higher chance to make more mistakes. We must use the percentage of a checked character appears in a practice text as a base (**O / P**).

A smaller P will make the MR larger. A larger P will make the MR smaller. The reason is that if a checked character appears a lot in a practice text, then the user has more opportunities to type it wrong. For example, in the same practice text which has 1000 checked characters. Letter "e" appears 50 times, so P(e) = 50 / 1000 = 0.05 and letter "w" only appears 5 times, P(w) = 5 / 1000 = 0.005. If a user types "e" wrong 10 times, but types "w" wrong 4 times. Obviously, this user should practice more "w" rather than "e", because MR(w) = 4 / 0.005 = 800 > 200 = 10 / 0.05 = MR(e).

5, Users can set how many practices results they want to keep for analyzing, for example 10 practices. After a user finishes the 11th practice, the first practice result will be deleted. The application will only keep 2nd to 11th practice results.

6. At any time, how to use a user's existing typing result to find out which practice text is the best for this user to improve typing skills? The principle is to find the practice test which has the highest checked characters appearance which this user has the highest mistake percentage.

7, Get the sum of "MistakeRatio" for each "CheckedCharacter" from the 10 saved typing result. Then sort the sum of "MistakeRatio" from high to low. For example, a user's first practice, the MistakeRatio for checked character "a" is MR(a)1. The second practice MistakeRatio for checked character "a" is MR(a)2. Then all the way to MR(a)10. If the user sets to save **n** practice results, then the formula is

$$MR(checked\ character)\text{sum} = \sum_{k=1}^{n} MR(checked\ character)k$$

where $MR = \frac{\text{O}}{\text{P}}$ and k is saved practice results order number

8, We select the top 3 MR sums to indicate 3 checked characters which have the highest chance to type wrong by this user, for example they are "h", "w" and "l", so users need to practice more of these characters. In the typing training application, users can set how many highest MR they want to show, for example, they might want to know which checked characters have the top 5 MR sums. Here we use 3 MR sums as example to show the algorithm.

9, Search through the saved practiced texts which belong to this user to find practice texts with the highest percentage of "h", "w" and "l". The logic is as the following

```
SELECT TOP (1) pt. Id, pt.Text
FROM CheckedCharacter cc INNER JOIN PracticeText pt ON cc.PracticeTextId = pt.Id
WHERE cc.Character = 'h' AND pt.UserId = 'currentUserId'
ORDER BY Percentage DESC
```

10, Repeat the previous logic for checked character “w” and “l”. Then suggest the 3 found practice texts to the user. It is possible that only 2 practice texts found because one text might have the highest percentage of two of the three checked characters “h”, “w” and “l”. When a text is has the highest percentage of all the three checked characters compared with other practice texts, then the user will only get one suggested practice text.

Note: This algorithm cannot prevent users from cheating. I know that type speed won’t be constant during typing, but I assume all users will do their best to type from the beginning to the end of a practice text, so the type results of how many times a character typed wrong can reflect which keys the user should practice more. However, for example, if a user knows he cannot type “b” very well, so he deliberately slows down a lot when types “b” or even look at the keyboard to avoid pressing a wrong key, then the algorithm won’t be able to suggest the best practice text.

2.9. Complexity

The technical complexity of this project consists of the follows:

Web server API: The Web API should handle the user authentication according to different roles.

Code first database control: it should handle multiple database schemas, for example there is a schema for user and role information, and another schema for users’ typing record related data. The Object Relational Mapping (ORM) used in this project is Microsoft Entity Framework Core, which is not the same as what I used previous pure Windows version Entity Framework.

User experience: this is another difficult part. The user interface should be attractive and intuitive, otherwise school kids won’t like it. The front web site must be responsive to both PC and mobile devices. Users can use mobile devices to login and see their typing reports. I need to learn Angular before start implementing this project.

Typing records analysis algorithm: This algorithm should efficiently calculate and produce a report for each individual user about the specific errors they typed and search existing stored text to suggest the best practice which target users’ weaknesses.

The management complexity of this project consists of the follows:

Risk management: There are some technologies I need to learn before using, for example Karma, the front-end unit test framework. I plan to use Angular Animation for the web interface. I also need to learn it first. Another risk is that this is an individual project, so I do not have teammates to discuss if I encounter difficulties and there will be no code peer review to find potential bugs.

Schedule management: This is the first time I develop such a big application individually. Some technologies I must learn. Some technologies I used before in my full-time job are based on senior developer’s setup and configuration, so I might encounter many troubleshooting problems to delay

the progress. My plan to solve this issue is to create some small test projects first to try these technologies, then start my major project.

Diploma students cannot solve this problem because they do not have the knowledge to individually manage and develop a project with such a scope and requirements. In BTECH's Technical Issues in Software Development and Management Issues in Software Development courses, I learned Agile methodology and what to do to prepare the risks before and during the development process.

Diploma students mostly focus on implementing the functionalities, but they do not have enough knowledge to think in detail about the software maintainability and development progress management. I graduated from CST program, so I know what I could do at that time. BTECH students can see the project from a high-level management point of view, for example use Agile Scrum methodology to keep delivering workable software. In one of my specialties, User Computer Interaction, we learned not just user interface (UI) design, but also user experience (UX) design principles. Sometimes a better user experience can make a software more competitive than just providing more functionalities. I also learned about SQLite, client-server access and responsive design in wireless and mobile course. These knowledges will be used in this typing software.

Diploma student do not pay too much attention to unit test in their projects. Most projects, at least in the CST program I took, students mostly use manual test. BTECH students can do much better in automated test to make the software more robust.

2.10. Research in New Technologies

This project uses the newest version of Microsoft .Net Core and Entity Framework to access SQLite database at the time I started this project. Dot Net Core has a build in dependency injection container. I encountered an issue that expected object cannot get instantiated by .Net Core dependency injection. Then after reading more documents, I figured out that the class I need to instantiated must first be registered in .Net code. I also figured out how to use Entity Framework to access SQLite database and its limitations.

This project uses the newest version of Google Angular and Bootstrap to build the web site front end. It also used the rxjs library to call .Net Core's Web API. I spent a lot of time to figure out the rxjs call to Web API must "subscribe" (a rxjs function) the call, otherwise the Web API won't get accessed. Each time I adopt new technologies, even though I think I understand how to use it, I still have to spend a lot of time on solving unexpected problems. This is the challenges and risk of using new technologies.

For the innovation component, I designed and implemented an algorithm to find the best practice texts saved in database according to users' previous typing records. I broke the algorithm into many steps and thought through each detail and potential problem, then refined them. The goal of the algorithm is to be correct, efficient and clear. During the implementation of this project, I realized that having an algorithm designed in detail is very helpful for implementation. Now I understand more deeply why the first time I handed in my project proposal; teachers required me to provide the algorithm steps in detail. Finally, I used actual tests to prove that my algorithm can find the best practice texts which targeting on users typing mistakes.

2.11. Future Enhancements

In the future, I can add a more advanced algorithm to analyze not just typing correctness of each single character, but also combo characters, such as “ear”, “ght”. For example, if a user often types letter “e” wrong, then the advanced algorithm can analyze the “e” which typed wrong appears in “ear”, or “ere” etc., so this typing application can suggest uses with better practice text to improve their typing skills more efficiently.

I can also add multiple language support in my online typing application. Users can set a preference language in account settings to select a language.

2.12. Timeline and Milestones

Online Typing Software Development	482 hrs	Mon 4/1/19	Tue 7/30/19
Sprint 1	**60 hrs**	**Mon 4/1/19**	**Mon 4/15/19**
Determine sprint scope	2 hrs	Mon 4/1/19	Mon 4/1/19
Create project template	2 hrs	Mon 4/1/19	Mon 4/1/19
Install & config 3rd party packages	4 hrs	Tue 4/2/19	Tue 4/2/19
Design data model	8 hrs	Wed 4/3/19	Thu 4/4/19
Implement data model for "Iam" schema	8 hrs	Fri 4/5/19	Sat 4/6/19
Data seeding	4 hrs	Sun 4/7/19	Sun 4/7/19
Web API "Iam" controller implementation	8 hrs	Mon 4/8/19	Tue 4/9/19
"Iam" service implementation	8 hrs	Wed 4/10/19	Thu 4/11/19
"Iam" repository implementation	8 hrs	Fri 4/12/19	Sat 4/13/19
Service side "Iam" test & debug	8 hrs	Sun 4/14/19	Mon 4/15/19
Sprint 2	**60 hrs**	**Tue 4/16/19**	**Tue 4/30/19**
determine sprint scope	2 hrs	Tue 4/16/19	Tue 4/16/19
Web UI design for user login	10 hrs	Tue 4/16/19	Thu 4/18/19
Routing for user login related pages	4 hrs	Fri 4/19/19	Fri 4/19/19
Front end user login service implementation	8 hrs	Sat 4/20/19	Sun 4/21/19
User login HTTP layer implementation	8 hrs	Mon 4/22/19	Tue 4/23/19
User login and account creation related test and debug	16 hrs	Wed 4/24/19	Sat 4/27/19
Typing interface design and implementation	12 hrs	Sun 4/28/19	Tue 4/30/19
Sprint 3	**60 hrs**	**Wed 5/1/19**	**Wed 5/15/19**
Typing logic implementation	20 hrs	Wed 5/1/19	Sun 5/5/19
Typing logic test and debug	12 hrs	Mon 5/6/19	Wed 5/8/19
Typing HTTP layer implementation	8 hrs	Thu 5/9/19	Fri 5/10/19
Back end implements data model for "Typing" schema	12 hrs	Sat 5/11/19	Mon 5/13/19
Back end Web API Typing controller implementation	8 hrs	Tue 5/14/19	Wed 5/15/19
Sprint 4	**60 hrs**	**Thu 5/16/19**	**Thu 5/30/19**
Back end typing service implementation	12 hrs	Thu 5/16/19	Sat 5/18/19
Back end typing repository implementation	12 hrs	Sun 5/19/19	Tue 5/21/19
Back end test and debug for typing	16 hrs	Wed 5/22/19	Sat 5/25/19
User typing report data model design	4 hrs	Sun 5/26/19	Sun 5/26/19
User typing report data model implementation	8 hrs	Mon 5/27/19	Tue 5/28/19
User typing report controller implementation	8 hrs	Wed 5/29/19	Thu 5/30/19
Sprint 5	**60 hrs**	**Sat 6/1/19**	**Sat 6/15/19**
User typing report service implementation	12 hrs	Sat 6/1/19	Mon 6/3/19
User typing report repository implementation	12 hrs	Tue 6/4/19	Thu 6/6/19
Back end user typing report test and debug	16 hrs	Fri 6/7/19	Mon 6/10/19
Front end typing report request service implementation	8 hrs	Tue 6/11/19	Wed 6/12/19
Front end typing report request HTTP layer implementation	8 hrs	Thu 6/13/19	Fri 6/14/19
Front end report request test	4 hrs	Sat 6/15/19	Sat 6/15/19
Sprint 6	**60 hrs**	**Sun 6/16/19**	**Sun 6/30/19**
Front end file upload UI implementation	8 hrs	Sun 6/16/19	Mon 6/17/19
Front end file upload service implementation	8 hrs	Tue 6/18/19	Wed 6/19/19
Front end file upload HTTP layer implementation	8 hrs	Thu 6/20/19	Fri 6/21/19
Back end file upload data model implementation	8 hrs	Sat 6/22/19	Sun 6/23/19
Back end file upload Web API controller implementation	8 hrs	Mon 6/24/19	Tue 6/25/19
Back end file upload service implementation	8 hrs	Wed 6/26/19	Thu 6/27/19
Back end file upload repository implementation	4 hrs	Fri 6/28/19	Fri 6/28/19
Back end file update test and debug	8 hrs	Sat 6/29/19	Sun 6/30/19
Sprint 7	**60 hrs**	**Tue 7/2/19**	**Tue 7/16/19**
Front end file loading UI implementation	4 hrs	Tue 7/2/19	Tue 7/2/19
Front end file loading service implementation	4 hrs	Wed 7/3/19	Wed 7/3/19
Front end file loading HTTP layer implementation	4 hrs	Thu 7/4/19	Thu 7/4/19
Back end file loading Web API controller implementation	4 hrs	Fri 7/5/19	Fri 7/5/19
Back end file loading service implementation	4 hrs	Sat 7/6/19	Sat 7/6/19
Back end file loading repository implementation	4 hrs	Sun 7/7/19	Sun 7/7/19
Back end file update test and debug	4 hrs	Mon 7/8/19	Mon 7/8/19
File loading test and debug	4 hrs	Tue 7/9/19	Tue 7/9/19
Typing error analysis algorithm implementation	20 hrs	Wed 7/10/19	Sun 7/14/19
System integration test	8 hrs	Mon 7/15/19	Tue 7/16/19
Sprint 8	**46 hrs**	**Fri 7/19/19**	**Tue 7/30/19**
Deployment to Azure	8 hrs	Fri 7/19/19	Sat 7/20/19
User manual	8 hrs	Sun 7/21/19	Mon 7/22/19
Final Report	30 hrs	Tue 7/23/19	Tue 7/30/19

Online Typing Software Development	482 hrs	Mon 4/1/19	Tue 7/30/19
◢ **Sprint 1**	**60 hrs**	**Mon 4/1/19**	**Mon 4/15/19**
Determine sprint scope	2 hrs	Mon 4/1/19	Mon 4/1/19
Create project template	2 hrs	Mon 4/1/19	Mon 4/1/19
Install & config 3rd party packages	4 hrs	Tue 4/2/19	Tue 4/2/19
Design data model	8 hrs	Wed 4/3/19	Thu 4/4/19
Implement data model for "Iam" schema	8 hrs	Fri 4/5/19	Sat 4/6/19
Data seeding	4 hrs	Sun 4/7/19	Sun 4/7/19
Web API "Iam" controller implementation	8 hrs	Mon 4/8/19	Tue 4/9/19
"Iam" service implementation	8 hrs	Wed 4/10/19	Thu 4/11/19
"Iam" repository implementation	8 hrs	Fri 4/12/19	Sat 4/13/19
Service side "Iam" test & debug	8 hrs	Sun 4/14/19	Mon 4/15/19
◢ **Sprint 2**	**60 hrs**	**Tue 4/16/19**	**Tue 4/30/19**
determine sprint scope	2 hrs	Tue 4/16/19	Tue 4/16/19
Web UI design for user login	10 hrs	Tue 4/16/19	Thu 4/18/19
Routing for user login related pages	4 hrs	Fri 4/19/19	Fri 4/19/19
Front end user login service implementation	8 hrs	Sat 4/20/19	Sun 4/21/19
User login HTTP layer implementation	8 hrs	Mon 4/22/19	Tue 4/23/19
User login and account creation related test and debug	16 hrs	Wed 4/24/19	Sat 4/27/19
Typing interface design and implementation	12 hrs	Sun 4/28/19	Tue 4/30/19
◢ **Sprint 3**	**60 hrs**	**Wed 5/1/19**	**Wed 5/15/19**
Typing logic implementation	20 hrs	Wed 5/1/19	Sun 5/5/19
Typing logic test and debug	12 hrs	Mon 5/6/19	Wed 5/8/19
Typing HTTP layer implementation	8 hrs	Thu 5/9/19	Fri 5/10/19
Back end implements data model for "Typing" schema	12 hrs	Sat 5/11/19	Mon 5/13/19
Back end Web API Typing controller implementation	8 hrs	Tue 5/14/19	Wed 5/15/19
◢ **Sprint 4**	**60 hrs**	**Thu 5/16/19**	**Thu 5/30/19**
Back end typing service implementation	12 hrs	Thu 5/16/19	Sat 5/18/19
Back end typing repository implementation	12 hrs	Sun 5/19/19	Tue 5/21/19
Back end test and debug for typing	16 hrs	Wed 5/22/19	Sat 5/25/19
User typing report data model design	4 hrs	Sun 5/26/19	Sun 5/26/19
User typing report data model implementation	8 hrs	Mon 5/27/19	Tue 5/28/19
User typing report controller implementation	8 hrs	Wed 5/29/19	Thu 5/30/19
◢ **Sprint 5**	**60 hrs**	**Sat 6/1/19**	**Sat 6/15/19**
User typing report service implementation	12 hrs	Sat 6/1/19	Mon 6/3/19
User typing report repository implementation	12 hrs	Tue 6/4/19	Thu 6/6/19
Back end user typing report test and debug	16 hrs	Fri 6/7/19	Mon 6/10/19
Front end typing report request service implementation	8 hrs	Tue 6/11/19	Wed 6/12/19
Front end typing report request HTTP layer implementation	8 hrs	Thu 6/13/19	Fri 6/14/19
Front end report request test	4 hrs	Sat 6/15/19	Sat 6/15/19
◢ **Sprint 6**	**60 hrs**	**Sun 6/16/19**	**Sun 6/30/19**
Front end file upload UI implementation	8 hrs	Sun 6/16/19	Mon 6/17/19
Front end file upload service implementation	8 hrs	Tue 6/18/19	Wed 6/19/19
Front end file upload HTTP layer implementation	8 hrs	Thu 6/20/19	Fri 6/21/19
Back end file upload data model implementation	8 hrs	Sat 6/22/19	Sun 6/23/19
Back end file upload Web API controller implementation	8 hrs	Mon 6/24/19	Tue 6/25/19
Back end file upload service implementation	8 hrs	Wed 6/26/19	Thu 6/27/19
Back end file upload repository implementation	4 hrs	Fri 6/28/19	Fri 6/28/19
Back end file update test and debug	8 hrs	Sat 6/29/19	Sun 6/30/19
◢ **Sprint 7**	**60 hrs**	**Tue 7/2/19**	**Tue 7/16/19**
Front end file loading UI implementation	4 hrs	Tue 7/2/19	Tue 7/2/19
Front end file loading service implementation	4 hrs	Wed 7/3/19	Wed 7/3/19
Front end file loading HTTP layer implementation	4 hrs	Thu 7/4/19	Thu 7/4/19
Back end file loading Web API controller implementation	4 hrs	Fri 7/5/19	Fri 7/5/19
Back end file loading service implementation	4 hrs	Sat 7/6/19	Sat 7/6/19
Back end file loading repository implementation	4 hrs	Sun 7/7/19	Sun 7/7/19
Back end file update test and debug	4 hrs	Mon 7/8/19	Mon 7/8/19
File loading test and debug	4 hrs	Tue 7/9/19	Tue 7/9/19
Typing error analysis algorithm implementation	20 hrs	Wed 7/10/19	Sun 7/14/19
System integration test	8 hrs	Mon 7/15/19	Tue 7/16/19
Sprint 8	**46 hrs**	**Fri 7/19/19**	**Tue 7/30/19**
Deployment to Azure	8 hrs	Fri 7/19/19	Sat 7/20/19
User manual	8 hrs	Sun 7/21/19	Mon 7/22/19
Final Report	30 hrs	Tue 7/23/19	Tue 7/30/19

3. Conclusion

This project helped me to improve my software developing skills from database, web server to front-end web site. This project will also include all the software planning, design and management skills I have learned from BCIT and the software development team in Schneider Electric. There might be some loopholes in my learning from school and work, but in this project, I found out and made up for these loopholes.

3.1. Lessons Learned

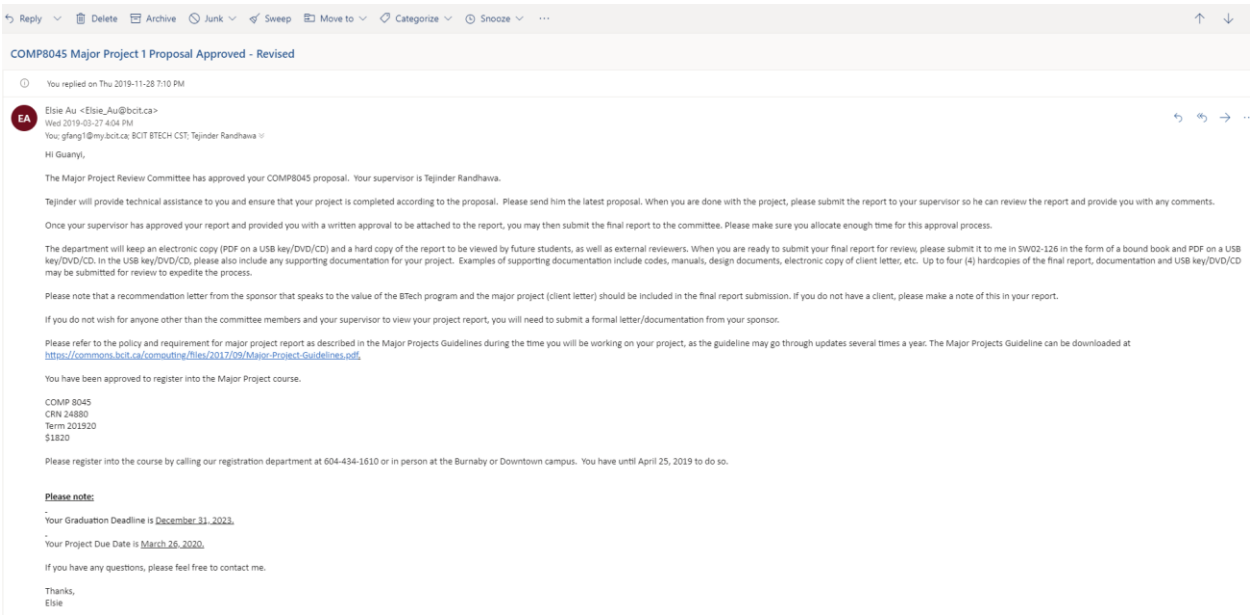
I should read more technical documents before choosing a technology in a project. During the development of my typing application, I found that the SQLite database does not support adding foreign keys by using Microsoft Entity Framework Core (EF Core) migration. According to my previous experience, developers can use EF Core to modify SQL Server database, but SQLite has more limitations then SQL Server. When I set the foreign key relation, I can only set it in the initial EF Core run, then if I need to add more foreign key relation later, I have to modify corresponding C# code and SQLite database separately. I realized that it's risky to adopt a new technology in a project.

3.2. Closing Remarks

This project is my last course to graduate from BETCH. Study in BCIT gives me a good career in IT industry. It changes my life. I would like to thank to BCIT teachers who help me, give me guidance during my study since CST to BETCH.

4. Appendix

4.1. Approved Proposal



Reply Delete Archive Junk Sweep Move to Categorize Snooze ···

COMP8045 Major Project 1 Proposal Approved - Revised

You replied on Thu 2019-11-28 7:10 PM

EA Elsie Au <Elsie_Au@bcit.ca>
Wed 2019-03-27 4:04 PM
You; gfang1@my.bcit.ca; BCIT BTECH CST; Tejinder Randhawa

Hi Guanyi,

The Major Project Review Committee has approved your COMP8045 proposal. Your supervisor is Tejinder Randhawa.

Tejinder will provide technical assistance to you and ensure that your project is completed according to the proposal. Please send him the latest proposal. When you are done with the project, please submit the report to your supervisor so he can review the report and provide you with any comments.

Once your supervisor has approved your report and provided you with a written approval to be attached to the report, you may then submit the final report to the committee. Please make sure you allocate enough time for this approval process.

The department will keep an electronic copy (PDF on a USB key/DVD/CD) and a hard copy of the report to be viewed by future students, as well as external reviewers. When you are ready to submit your final report for review, please submit it to me in SW02-126 in the form of a bound book and PDF on a USB key/DVD/CD. In the USB key/DVD/CD, please also include any supporting documentation for your project. Examples of supporting documentation include codes, manuals, design documents, electronic copy of client letter, etc. Up to four (4) hardcopies of the final report, documentation and USB key/DVD/CD may be submitted for review to expedite the process.

Please note that a recommendation letter from the sponsor that speaks to the value of the BTech program and the major project (client letter) should be included in the final report submission. If you do not have a client, please make a note of this in your report.

If you do not wish for anyone other than the committee members and your supervisor to view your project report, you will need to submit a formal letter/documentation from your sponsor.

Please refer to the policy and requirement for major project report as described in the Major Projects Guidelines during the time you will be working on your project, as the guideline may go through updates several times a year. The Major Projects Guideline can be downloaded at https://commons.bcit.ca/computing/files/2017/09/Major-Project-Guidelines.pdf.

You have been approved to register into the Major Project course.

COMP 8045
CRN 24880
Term 201920
$1820

Please register into the course by calling our registration department at 604-434-1610 or in person at the Burnaby or Downtown campus. You have until April 25, 2019 to do so.

Please note:

Your Graduation Deadline is December 31, 2023.

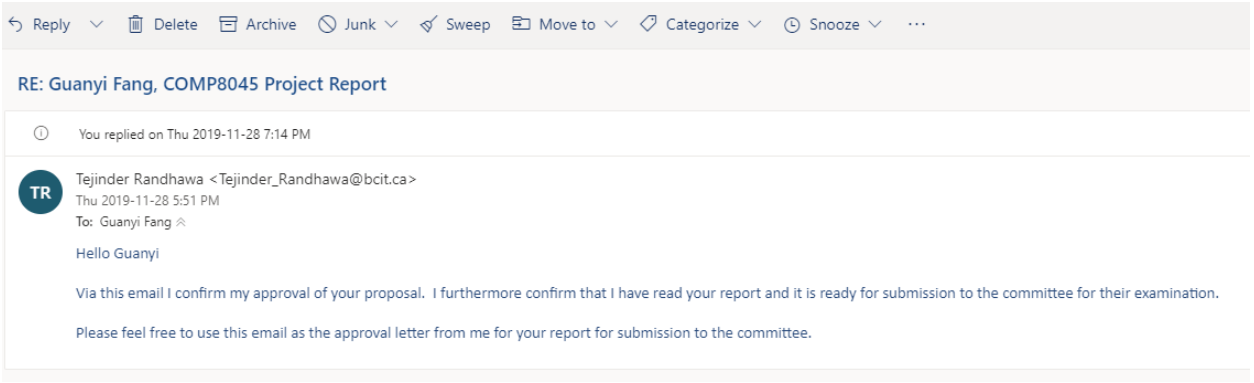
Your Project Due Date is March 26, 2020.

If you have any questions, please feel free to contact me.

Thanks,
Elsie

4.2. Project Supervisor Approvals

My project supervisor is Tejinder Randhawa. He approved my report on Nov 28th, 2019.



Reply Delete Archive Junk Sweep Move to Categorize Snooze ···

RE: Guanyi Fang, COMP8045 Project Report

You replied on Thu 2019-11-28 7:14 PM

TR Tejinder Randhawa <Tejinder_Randhawa@bcit.ca>
Thu 2019-11-28 5:51 PM
To: Guanyi Fang

Hello Guanyi

Via this email I confirm my approval of your proposal. I furthermore confirm that I have read your report and it is ready for submission to the committee for their examination.

Please feel free to use this email as the approval letter from me for your report for submission to the committee.

5. References

- Angular framework https://angular.io/
- TypeScript https://www.typescriptlang.org/
- Bootstrap https://getbootstrap.com/
- Node.js https://nodejs.org/en/
- NPM https://www.npmjs.com/
- ASP.NET Core https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-2.2
- Entity Framework Core https://docs.microsoft.com/en-us/ef/core/
- AutoMapper https://automapper.org/
- Moq https://github.com/moq/moq4
- SQLite https://www.sqlite.org/index.html
- Azure https://azure.microsoft.com/en-ca/

- Jira https://www.atlassian.com/software/jira
- Bitbucket https://bitbucket.org/product
- Sourcetree https://www.sourcetreeapp.com/

6. Change Log

Version 2 on January 14th, 2020 according to feedback.

- Section 1.1 Student Background, add more education and previous work experience.
- Section 1.2 Project Description, add more description of this project.
- Section 2.3 Possible Alternative Solution and Section 2.4 Chosen Solution, change to technical solution choice instead of conceptual.
- Section 2.5 Detail of Design and Development, add more text description for each diagram.
- Section 2.6 Test Detail and Results, add description for each unit test and explain of the manual test process according to the diagram. Rename section 2.6.2 to Manuel Feature Test.
- Section 2.8 Innovation, move the general description of the project from section 2.8 to section 1.2.