CSC 431

Focus Time

System Architecture Specification (SAS)

Team 14

Felipe Flores	Scrum Master
Tianyu Ma	Developer
Wen Li	Developer

Version History

Version	Date	Author(s)	Change Comments
1	3/30/2022	Felipe Flores, Tianyu Ma, Wen Li	First draft

Table of Contents

Table of Figures

1. System Analysis

1.1 System Overview

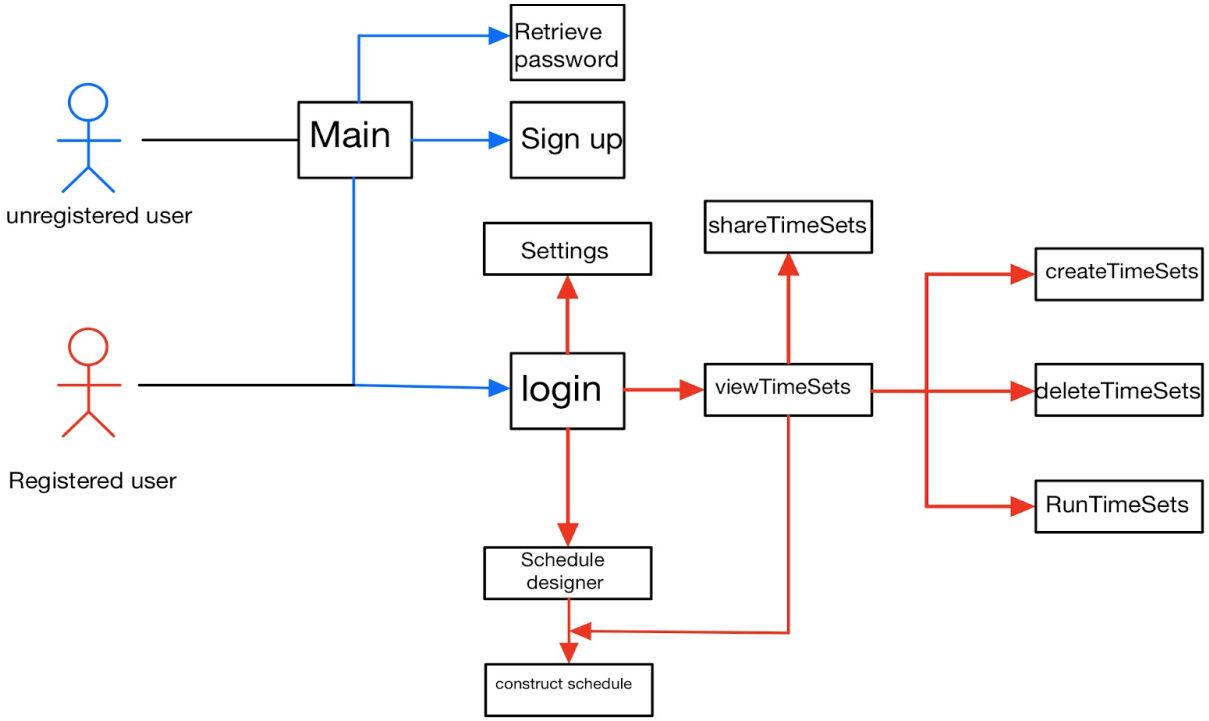
Our system is composed of the following main parts: Login Form/Registration Form, Database, Account, UI, TimeSet Manager, and TimeSet.

The Login Form/Registration Form, Database, and Account are all responsible for allowing all the functionalities related to the user's access to their account. The Forms are what the user interacts with to input their information to login or register into our system as well as alerting the user if the request was successful. The Database contains all previously created accounts within our system as well as verifying any new incoming credentials from the user with its stored data. The Account serves as a set of data, containing an email and password. Collectively, these parts enable the user's ability to log in and register into our system.

The UI, TimeSet Manager, and TimeSet are all responsible for allowing all the functionalities related to the user's potential interactions with their TimeSets. The UI serves to display information about TimeSets to the user, as well as alerting the user if their desired action was successful. The TimeSet Manager is the caretaker of all TimeSets within our system. It is responsible for storing TimeSets locally within the user's device, modifying a TimeSets parameters to the user's liking, and executing a selected TimeSet per the user's request. These enable the user's functionality of creating, accessing, and sharing TimeSets. The TimeSet serves as a set of data, containing timer duration, block strength, and a list of music files. All these parts provide the main blocking service our app strives for.

Our selected architecture design will implement a three tier architecture given how our system encompasses functional process logic, data access/storage and a user interface.

1.2 System Diagram



Above is an overview of the entire system in one diagram. All functionalities are accessible once users are registered. If users don't want to share personal information, they can register without an email address. And hence they won't be able to retrieve their password

1.3 Actor Identification

1. Student/worker: It is a user(human) type of actor. A user registers on the applications' use interface to access/add/delete/share TimeSets. The user is our primary actor because he/she initiates the interaction of the system.
2. Administrator: it is a system type of actor. This actor, from the external system,is a secondary actor.
3. Time: It is a system clock. It is the time clock of users' mobile devices to monitor the study time of users.
4. Speaker: It is a system speaker. It is the speaker of users' mobile devices to play music during TimeSets.

1.4 Design Rationale

1.4.1 Architectural Style

The app uses a three-tier architectural style. The first tier is the presentation tier. The second tier is the application tier. The third tier is the data tier.
The presentation tier is similar to the user interface and communicates with the other two tiers in the system. It displays users' Timeset and collects their Timeset through a web feature.
In the application tier, it collected the information from the presentation tier. Specifically, it uses business logic to maintain the app's logic. It can add, delete or modify the data.
In the data tier, it kept the database to store users' data. It is where the information is stored and managed in TimeSet.
In this three-tier design, the application tier communicates between the presentation tier and data tier. The presentation tier has no direct connection with the data tier.

1.4.2 Design Pattern(s)

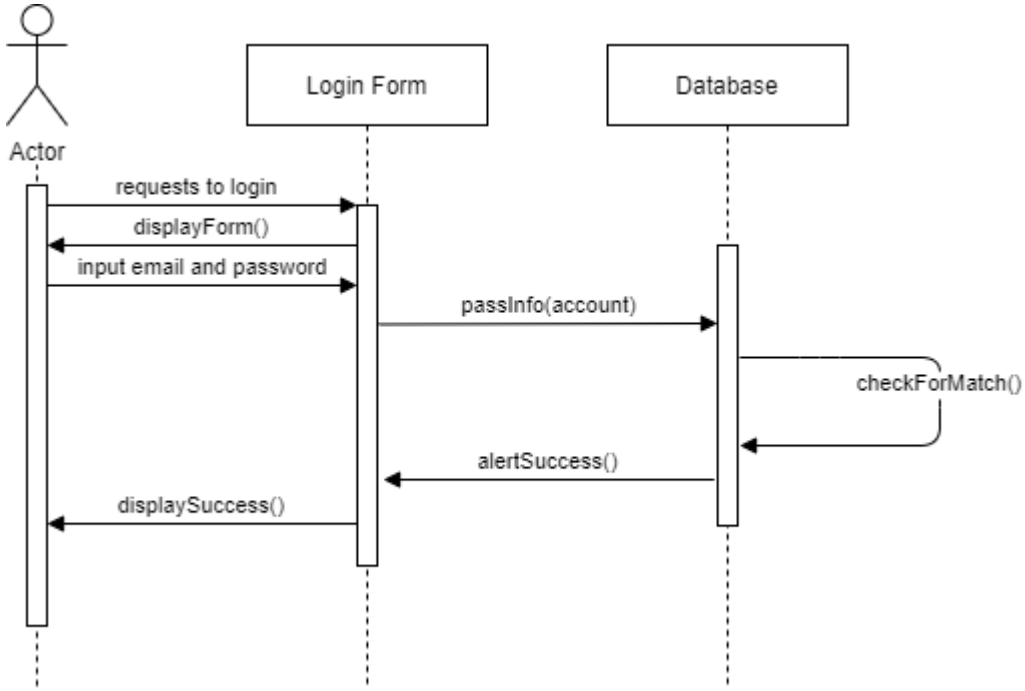
The creation design pattern we use is the Factory Method. Factory Method Creates an instance of several derived classes. It has a particular format in different classes. In Timeset, a user can create his TimeSet by putting the information he wants to save. When users access the Timeset, the data is collected from all classes. The information is accessed quickly and well-formatted in structure, making it easier for users.
Since some of the data in Timeset is encrypted, the private class data can restrict certain accessors from accessing the data.
To ensure the Timeset is saved and uploaded to the database, a mediator can define the simplified communication between classes to check if anything is left behind.

1.4.3 Framework

Our mobile framework is Xamarin. Xamarin is an open-source app platform from Microsoft for building IOS and Android apps with C#. Since the app will apply on the phone, we need specific libraries to track the time users spend on their apps.
For the web version, our framework is Angular UI grid. Angular UI grid is one of the Angular frameworks used to deal with large datasets. Angular UI Grid is good at sorting, grouping, and user interaction. All of these features will help users better access their Timeset. As developers, it is easier for us to control user-saved Timeset.

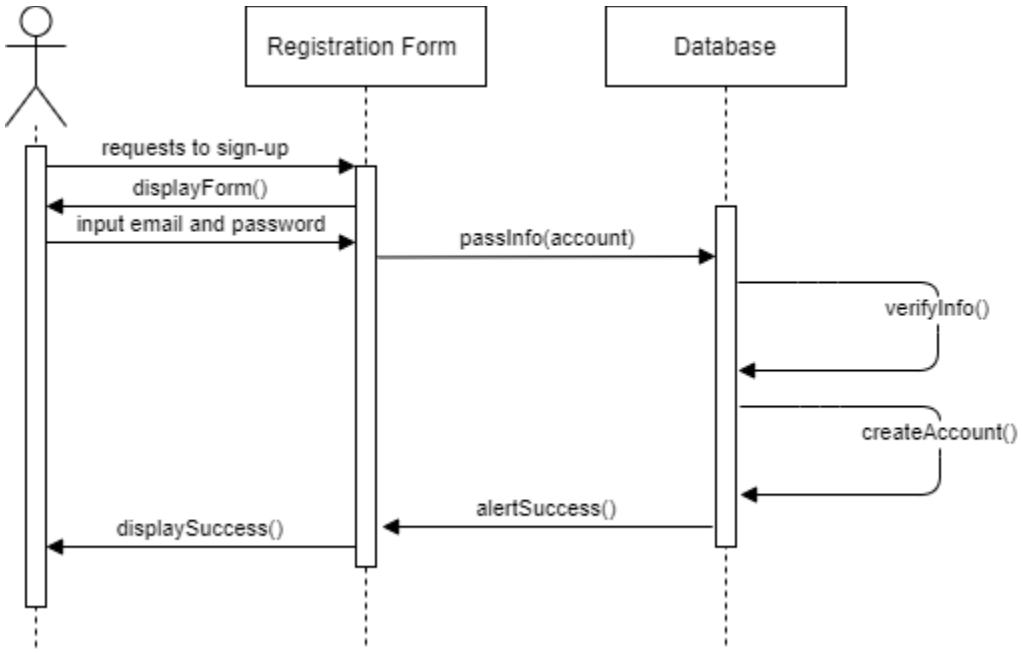
2. Functional Design

2.1 Log in Sequence Diagram



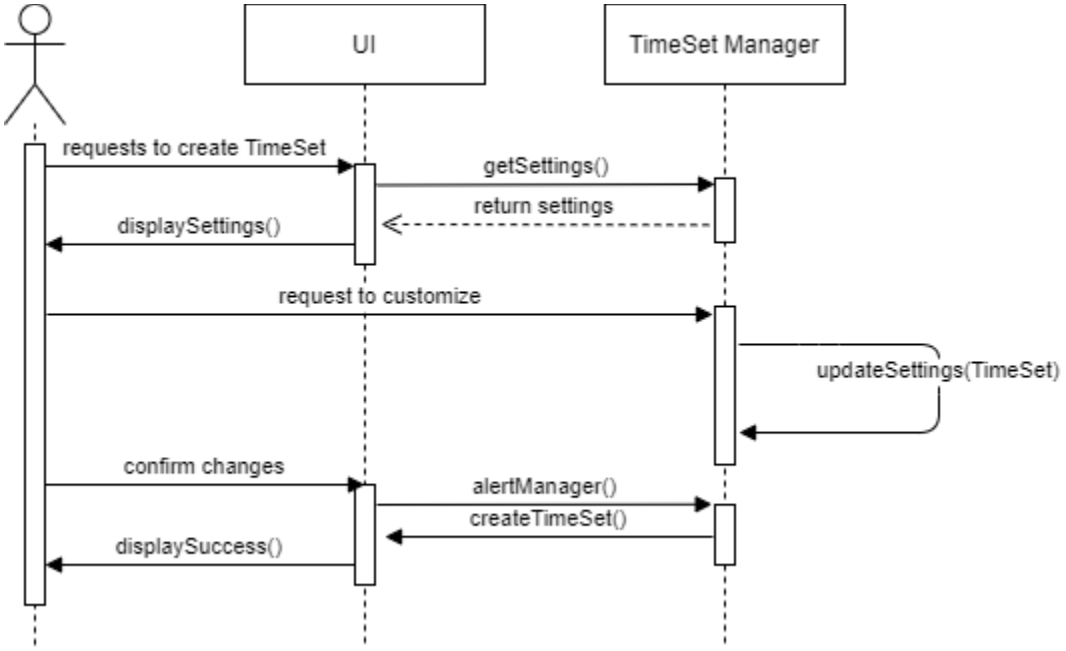
- The User first requests to log into our system, to which the Login Form responds by displaying the form to log in
- Then the User inputs their email and password onto the Login Form.
- The Login Form passes this information to the Database.
- Then the Database compares this information to all the accounts it has stored to search for a match.
- Should the database find a match, then it will alert the Login Form that the log in is a success.
- The Login Form in turn displays the success to the User, and allows to the user to access the rest of our system's functionalities

2.2 Sign up Sequence Diagram



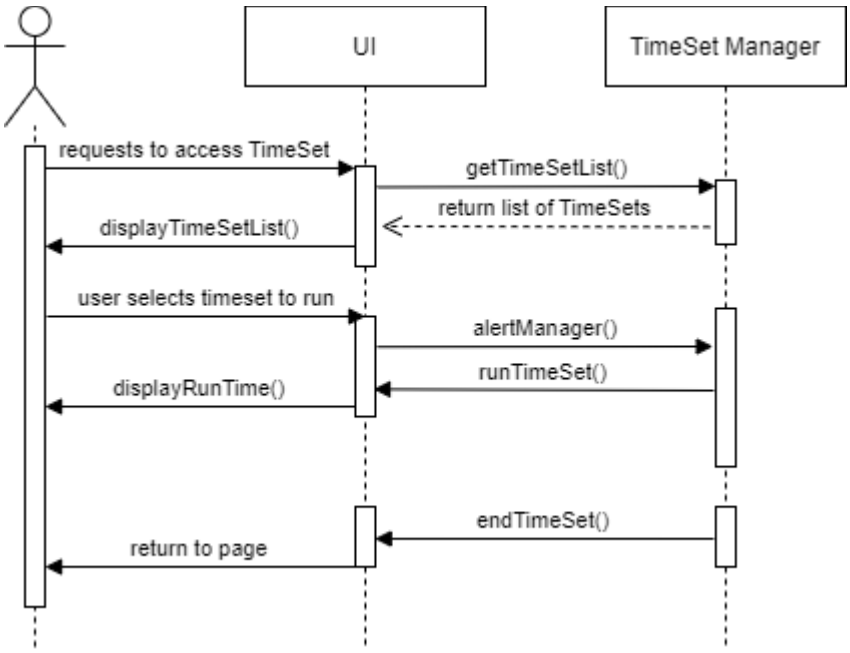
- The User first requests to sign up into our system, to which the Registration Form responds by displaying the form to log in
- Then the User inputs their email and password onto the Registration Form.
- The Registration Form passes this information to the Database.
- Then the Database verifies this information to see if the email given exists and the password meets specific requirements.
- Should the Database verification be successful, then the Database will create a new account within our system.
- The Database alerts the Registration Form to the success of the creation of a new account.
- The Registration form then displays the successful creation of a new account to the user.

2.3 Create TimeSet



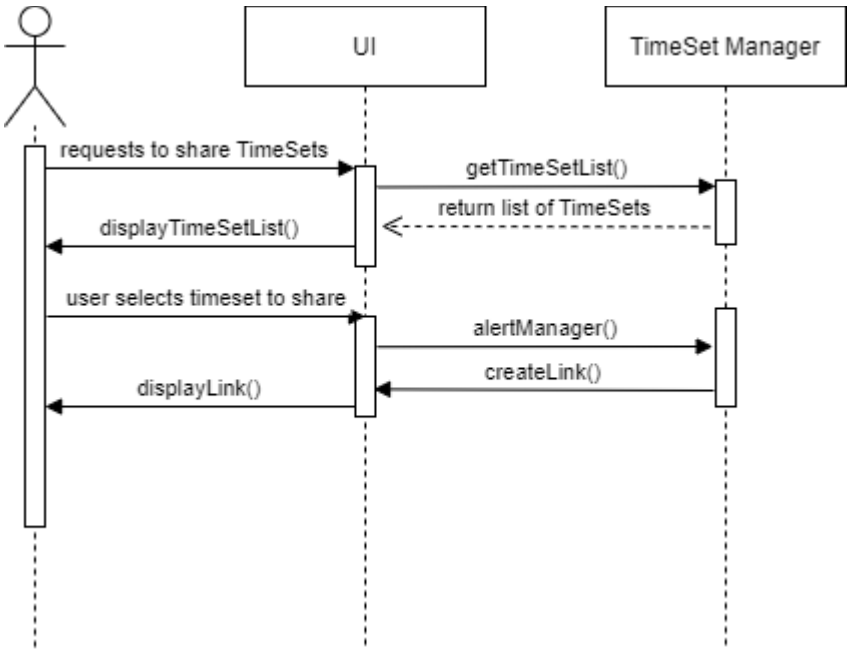
- The User first requests to create a new TimeSet in our system to the UI
- The UI retrieves the customizable options of a TimeSet from TimeSet Manager
- The UI then displays the customizable options to the User.
- The User requests to customize the options presented to them, changing the options to their liking.
- The TimeSet Manager takes the requested changes and updates the TimeSet accordingly.
- The User confirms these changes in the UI
- The UI alerts the TimeSet Manager of this confirmation, and the TimeSet Manager creates the TimeSet with the finalized options
- The UI displays to the User that the creation was successful

2.4 Access TimeSets



- The User first requests to access TimeSets in our system to the UI
- The UI retrieves the list of all TimeSets from the TimeSet Manager
- The UI then displays the all the TimeSets to the User in a list.
- The User selects a TimeSet to run
- The UI alerts the TimeSet Manager and the TimeSetManager begins to run the requested TimeSet
- The UI displays the remaining runtime of the TimeSet to the User
- When the runtime ends, the TimeSetManager ends the TimeSet, and the UI returns the user to the list of TimeSets

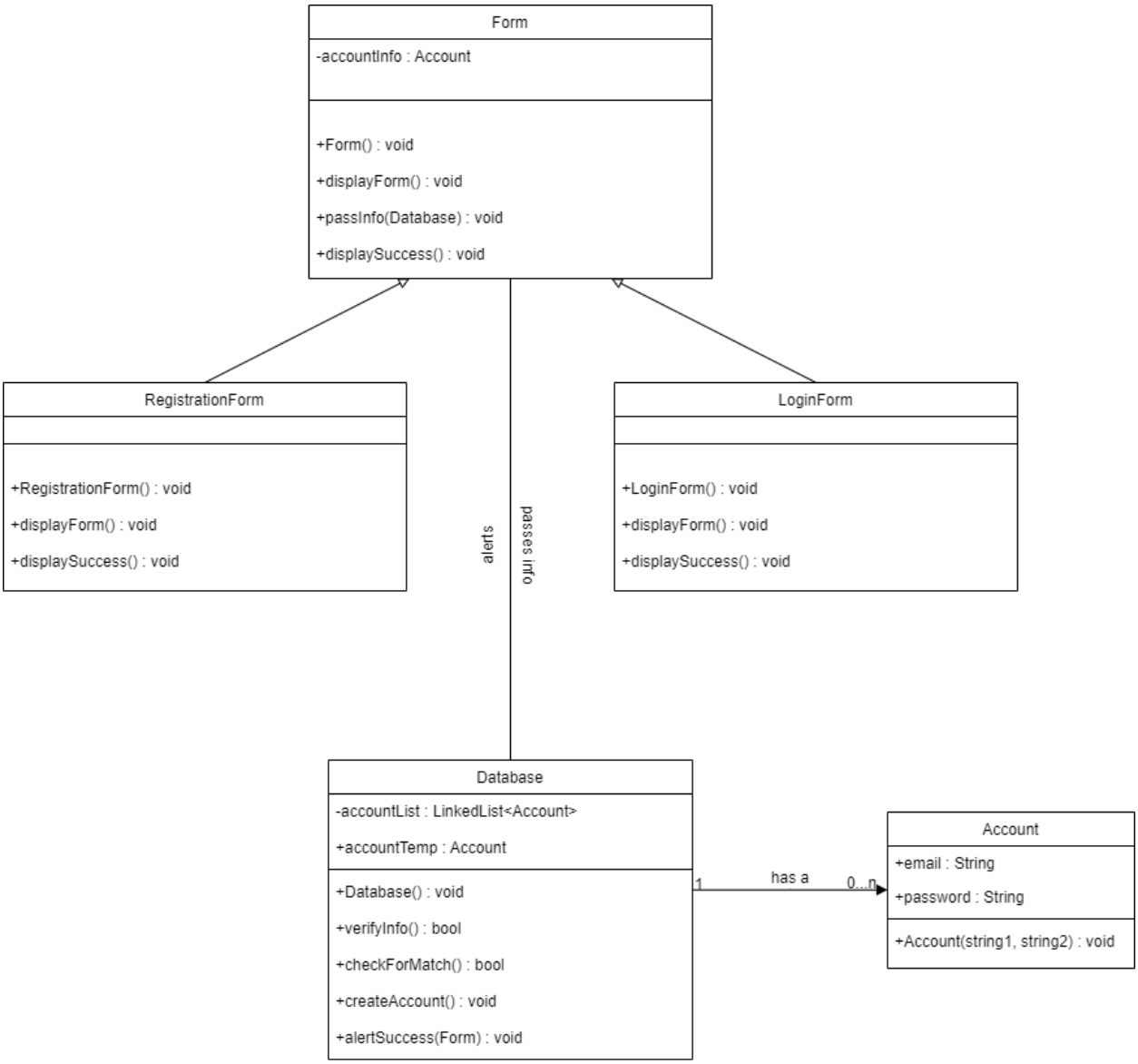
2.5 Share TimeSets



- The User first requests to share TimeSets in our system to the UI
- The UI retrieves the list of all TimeSets from the TimeSet Manager
- The UI then displays the all the TimeSets to the User in a list.
- The User selects a TimeSet to share
- The UI alerts the TimeSet Manager and the TimeSet Manager creates a link of the requested TimeSet
- The UI displays this link to the User
- The User can then share copy this link and distribute by means outside of our system

3. Structural Design

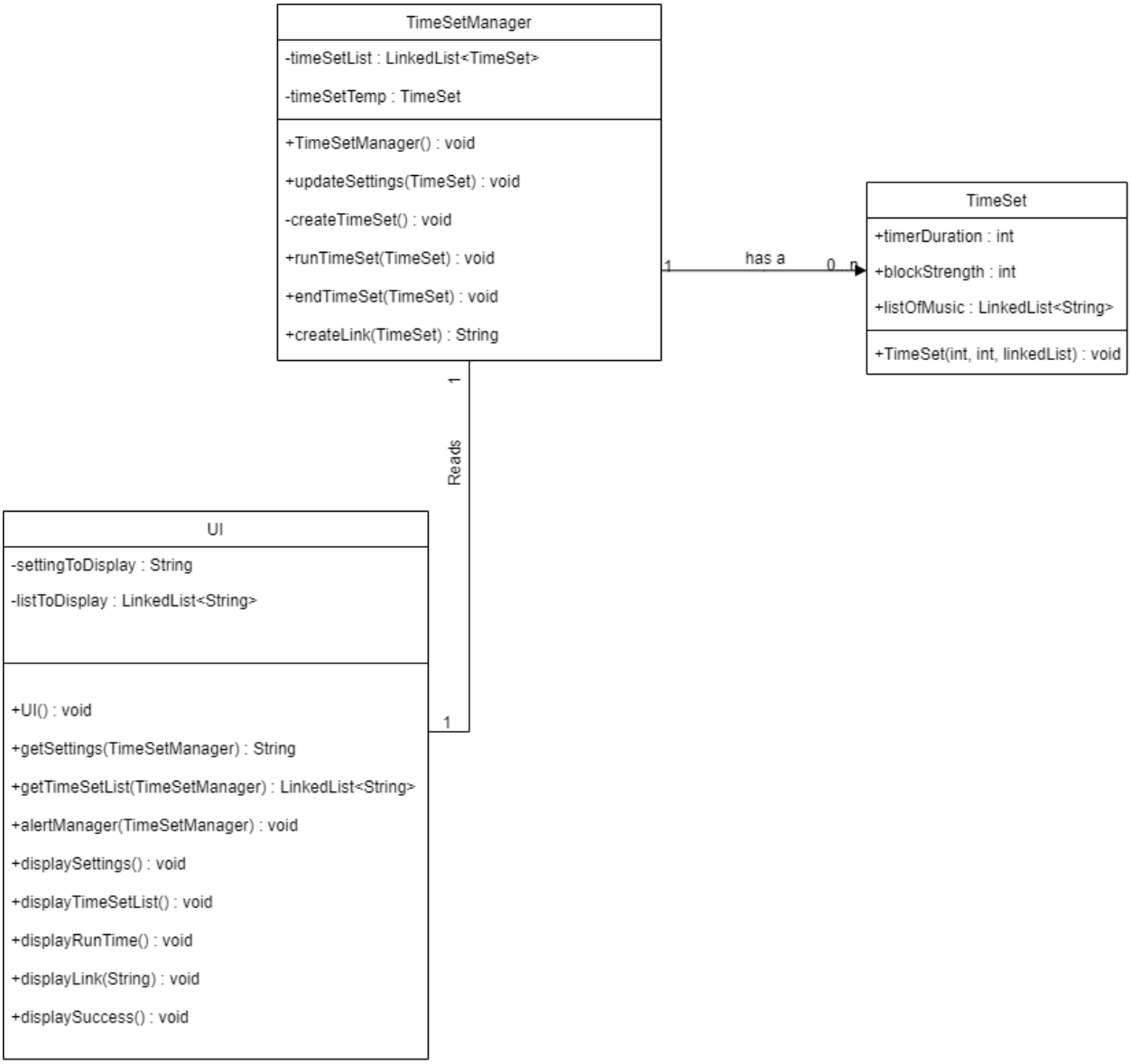
3.1 UML Class Diagram 1(Form, Database, Account)



- The Form parent class defines the base behavior of the systems two main forms: Login Form, and Registration Form.
- Login Form and Registration Form have slightly different behaviors, these varying behaviors are the ones specified in the diagram.
- Both Forms pass information to the Database, to which the Database responds with alerts back to the form to then alert the user that the operation was a success
- The Database contains a list of accounts as well as a temporary account. This temporary account serves to store the incoming account information passed by the Forms.

- The Account class is going to be defined as struct within our system, as its only purpose is providing a convenient way to store the set of data for an account. Those being email and password.

3.2 UML Class Diagram 2(UI, TimeSet Manager, TimeSet)



- The UI class serves as the main gateway for the User to interact with the rest of the system, as well as reading data that the TimeSet Manager contains to display said data to the User.
- The TimeSet Manager is in charge of containing the data related to the User's TimeSets, updating any TimeSets, creating new TimeSets, running TimeSets, and creating links from TimeSets.
- The timeSetTemp variable in our TimeSet Manager serves to contain incoming TimeSet data from UI
- The TimeSetManager contains many TimeSets, as seen from the "has a" relationship
- The TimeSet class is going to be defined as a struct within our system, as its only purpose is to provide a more convenient way to store the set of data for a TimeSet. Those being the duration of the timer, block strength, and a list of music files.