**Documentation**

The front end of our application was written in google dart using a framework called flutter. Flutter is a frontend app development framework that can be used to make cross platform apps (i.e. Mobile and Desktop).

**General Repository Overview**

The front end of our application resides in the UI folder of registration magic repository.Graphical user interface, application

Description automatically generated

The lib directory houses the code implemented by our group for the applications front end user interface.

Graphical user interface, text, application

Description automatically generated

**Front End Implementation**

Graphical user interface, application

Description automatically generated

Our lib directory contains the **main.dart** main method for our application as well as three folders that house the viscera of our project.

**Database**

Graphical user interface, application, Teams

Description automatically generated

The Data folder contains database.dart : an sql database which stores the list of classes that our user will register.

The following functions are stored in database.dart:

* void createInitialData()

This function initializes the database and fills class list with default classes.

* void loadData()

This function loads the previously stored data in classList.

* void setTime(String time)

This function sets the desired registration time

* void loadTime()

This is a helper function that retrieves the stored registration time from the database.

* void updateTime()

This updates the time in the database with the new time input by the user.

* void updateDataBase()

This function updates the database with changes made to the list of classes to be registered.

* String returnJsonList()

This function returns the classList as a JSON string.

* String returnLatestTime()

This function returns the classlist as a string.

**Screens**

Flutter applications can house multiple screens. The screens directory houses the login and home pages as well as a widgets folder which houses other assets (custom widgets in flutter terms). Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, Teams

Description automatically generated

**Util**

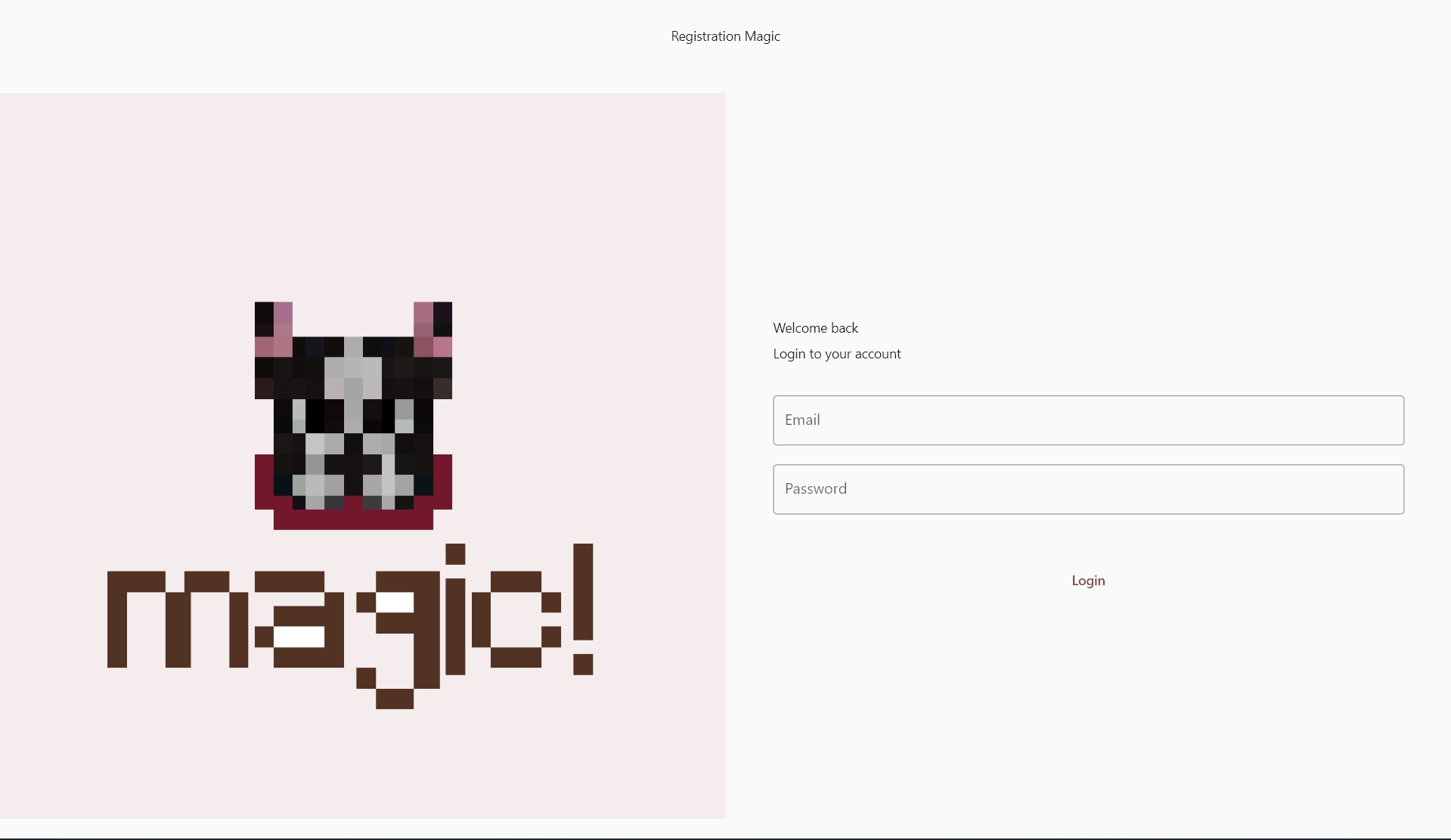
Table

Description automatically generated

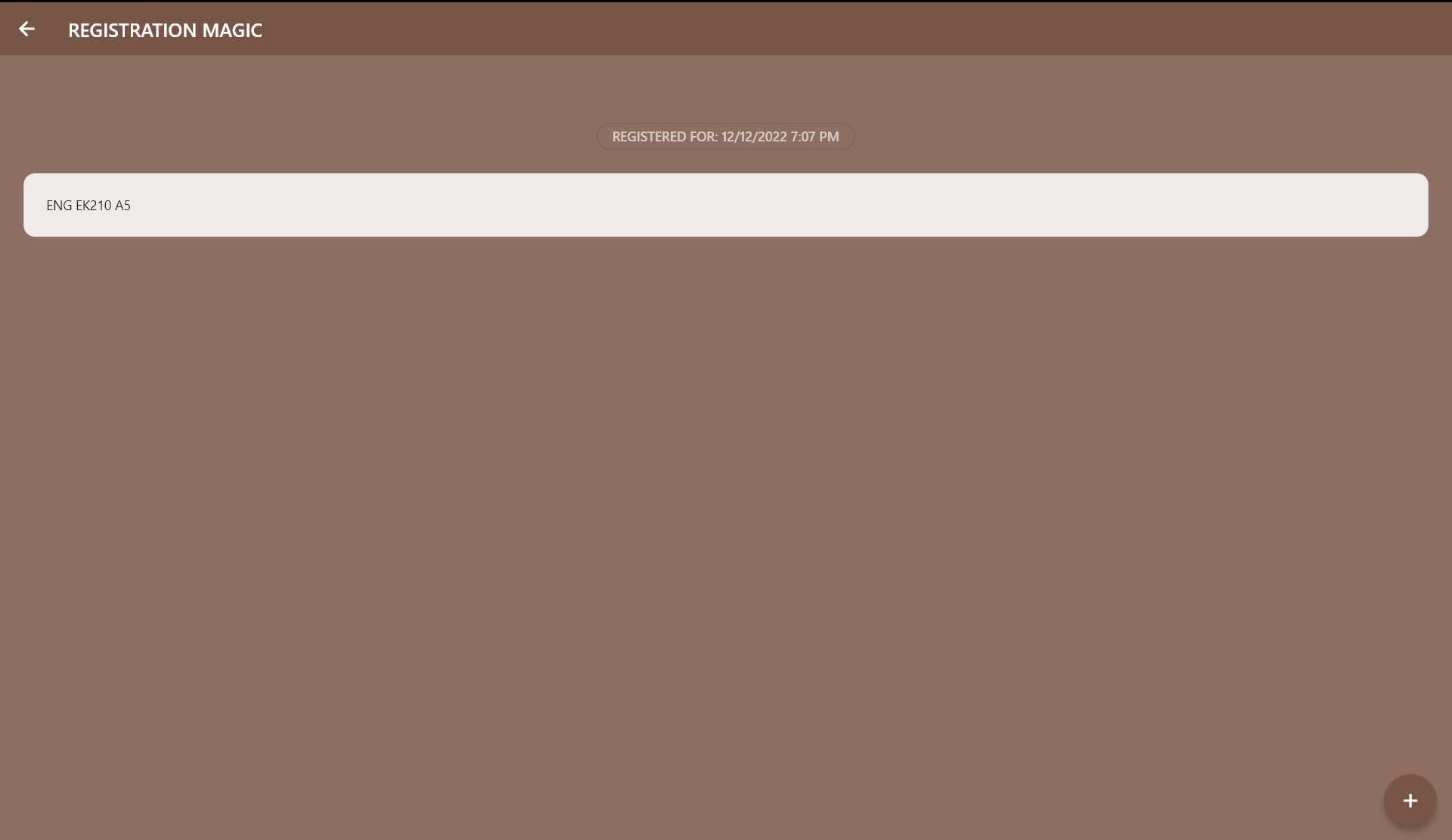
Finally, the Util folder holds another set of utility widgets.

**Application Screenshots:**

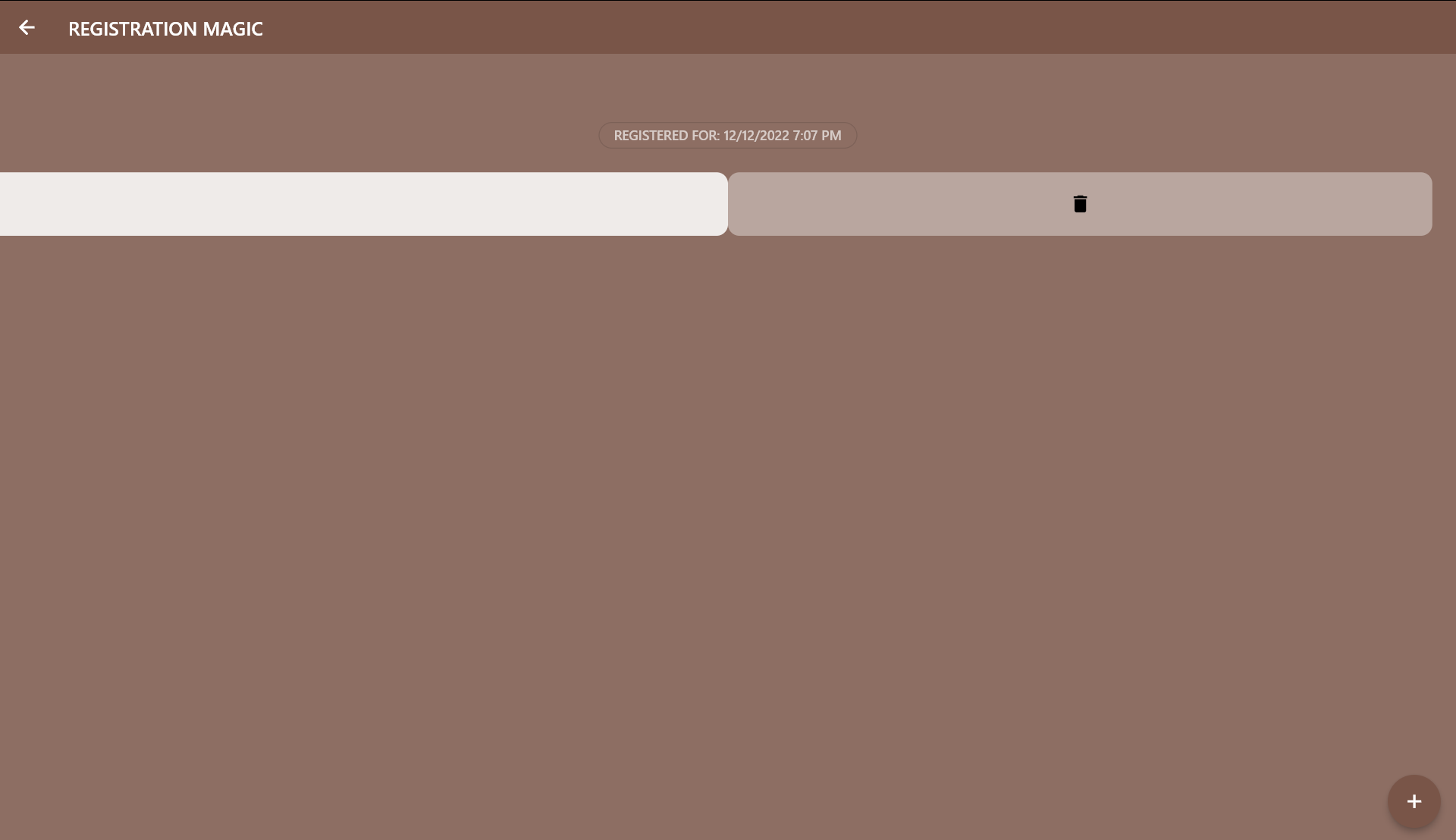
**Login Page**

****

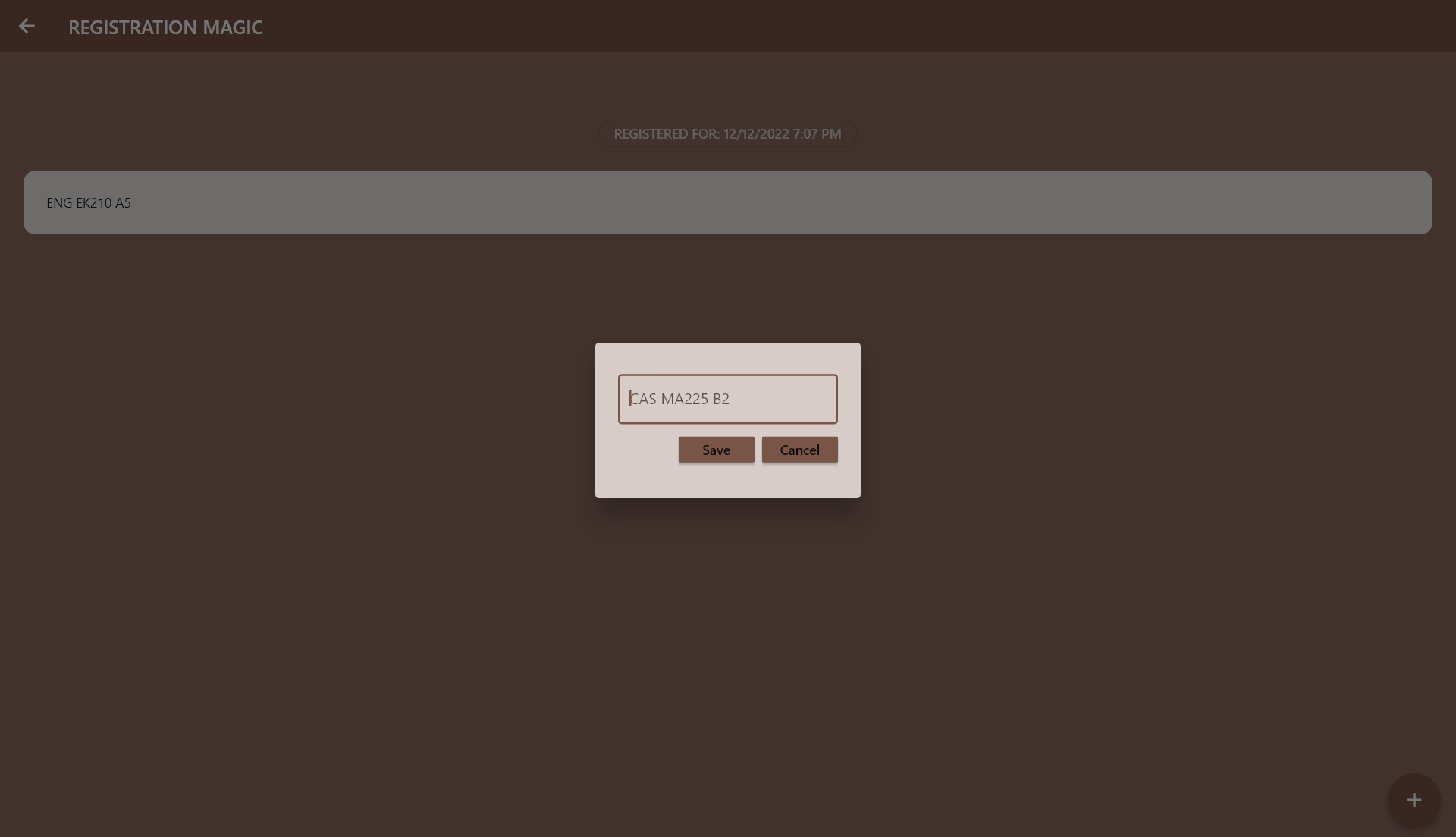
**Home Page**

****

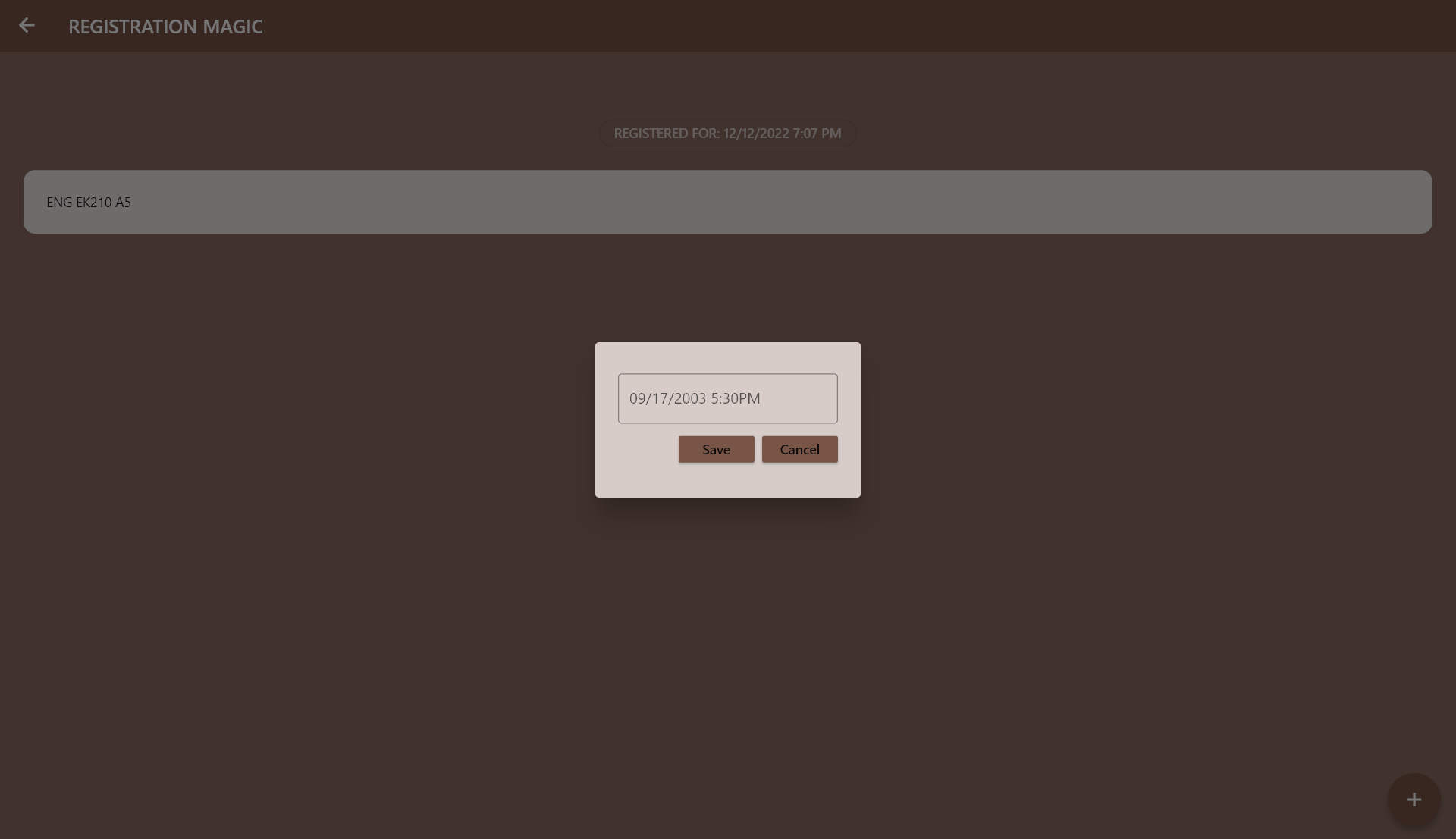
**Class Deletion Functionality**

****

**Add a Class**

****

**Set Registration Time**

****

**Backend**

main.py

The backend uses asynchronous tasks running a Flask app and the registration functions as scheduled at specified times. The Flask app acts as a server receiving the API tokens sent from the frontend, which are passed to functions that add the relevant info to a User which is then executed with user.register().

courses.py

This contains the data structures used in the rest of the backend, including the course itself, which is defined thusly:

college: a 3-letter string defining the college, e.g. 'ENG'

department: a 2-letter string defining the department, e.g. 'EC'

course\_num: an integer defining the course number, e.g. 327

section: alphanumeric string defining the section

coursedb.py

This reads from the Courses\_Info.xlsx file, which contains all the data scraped from the Student Link by scraper.py

user.py

This contains the courses for each semester, and a register function that takes this information and turns it into a request that captures the kerebos auth token and then registers the user for the requested courses.