The Four Year Planner Readme

Running Our Program

We suggest running the code on **Visual Studio Code**, because we have never tried running it on any other IDE. We are assuming that Flutter is already installed in the user's computer.

Note: We have made a demo video as suggested by Barbara from our experience debugging the code during office hours on Monday. This is the link to it. The passcode is tJyn=e8p

The steps to run are as follows:

1) Clone the GitHub repository from the course's GitHub Classroom
2) Ensure that you have flutter and dart extensions installed in VSCode
3) Run 'flutter pub get' in the Terminal window
4) Next run 'flutter pub add googleapis' in the Terminal window
5) Even though the repository has the latest pubspec.yaml, ensure it matches with this:

```
version: 1.0.0+1

environment:
  sdk: ">=2.1.0 <3.0.0"

dependencies:
  flutter:
    sdk: flutter
  syncfusion_flutter_calendar: ^18.3.40
  intl: ^0.16.0
  google_sign_in: ^4.4.6
  googleapis_auth: ^0.2.11+1
  googleapis: ^0.54.0
  cupertino_icons: ^0.1.2
  url_launcher: ^6.0.3
  percent_indicator: ^2.1.7+2

dev_dependencies:
  flutter_test:
    sdk: flutter

flutter:
  uses-material-design: true
  assets:
    - assets/
```

- assets/gates.jpg

6) The lib folder of the flutter project should have no errors now. We are all set to run the server.
7) Running the server:
    a) Type *ssh <username>@cs-vm-01.cs.mtholyoke.edu*
    b) Clone the project from GitHub
    c) Navigate to lib folder
        1. *cd group_project_4_year_planner*
        2. *cd lib*
    d) In lib folder, type: *node server.js*
    e) Press Ctrl + C to stop the server
8) Once the server is set up, go to lib > main.dart and press the run button (the triangle on the top left corner of the screen). Ensure you have a device selected. We recommend an emulator device, but if speed is an issue you can create a web browser by running 'flutter create .' on the Terminal and choosing 'Chrome'.
9) The output will be on the screen

Roadmap of the Directory Structure:

The core of the project is in the lib folder. We have the following .dart files in lib with their role mentioned:

1) home.dart:

    This file contains the class Home. This class builds a home page which welcomes the user to the and instructs them on some of the functionality. The app will always open to this page when started up.

2) GoogleCalendar.dart:

    This file contains three classes: GoogleEvents, GoogleDataSource, and GoogleAPIClient. GoogleEvents contains static methods which access and store the events of each of the calendars in their respective list. These lists can be found in globals.dart.

    GoogleDataSource contains a constructor which takes a list of Events as an argument and assigns this list to its field appointments. This structure and the class's functions allows the SFCalendar widget to take a GoogleDataSource object and project its events.

    GoogleAPIClient is a basic class that constructs an object which allows for the connection in GoogleEvents to be made.

3) academicPlanner.dart:

This file contains the class AcademicPlanner which allows users to select their major. Upon selection, a checklist is projected which asks the user to specify which courses they have already completed.

4) Calendar.dart:

   This file contains the class CalendarEvents which uses the SyncFusion calendar package to project deadlines relevant to the users extracurricular interests in a schedule format.

5) picCard.dart:

   This file contains two classes, PicCard and Deadlines. PicCard takes as arguments relevant information to extracurriculars and builds a widget which displays this information in a neat card. These cards make up the Beyond the Classroom page.

   Deadlines takes as an argument a String calendar which specifies a List of Events and builds a widget which lists all events of that calendar. These lists are embedded within each PicCard.

6) beyondClassroom.dart:

   This file contains the class PicCards, which builds a widget containing a PicCard for each extracurricular. PicCards is the main widget for the Beyond the Classroom page.

7) Classinfo.dart:

   This file contains the class ClassInfo which is meant to display to the user their course completion history as part of the academic planning section. However, it is not yet fully implemented.

8) courseTiles.dart

   This file contains the class CourseTiles which builds a widget that functions as a checklist for major classes only as part of the academic planning section.

9) distributionReqs.dart

   This file contains the class CourseTiles which builds a widget that functions as a checklist for distribution requirements only as part of the graduation requirements section.

10) globals.dart

This file contains all global variables used in this program.

11) requirements.dart

This file contains the class Requirements which builds the Graduation Requirements page. This page contains two progress indicators, as well as the distribution requirement checklist implemented in distributionReqs.dart.

12) drawer.dart

This file makes the navigation menu work. It contains links to different files so that we can switch between them while using the app.

13) drawer2.dart

This file contains the class DrawerNavigation which builds a drawer widget allowing the user to switch from page to page. This widget is visible from Home, Beyond the Classroom, and Schedule. When the user is in the Academic Planning page, DrawerNavigation2 is used instead so that more options can be shown.

14) main.dart

This file contains the class DrawerNavigation2 which builds a drawer widget allowing the user to switch from page to page. This widget is visible from all pages pertaining to Academic Planning. When the user is in Home, Schedule, or Beyond the Classroom, DrawerNavigation is shown instead so as to hide the other academic pages.

Functionalities Implemented:

1) Minimum Viable Product (MVP):
    a) Providing information about the extracurricular activities offered by the college. This is implemented as the "Beyond the Classroom" page.
    b) Allowing the user to choose the activities they wish to participate-in. This is implemented as a part of the checklists in the "Beyond the Classroom" page.
    c) Scheduling extracurricular activities outside of academics to give visualization of the deadlines. This is implemented in the "Schedule" page. The deadlines displayed in this page are the ones for which the user has checked boxes in the "Beyond the Classroom" page.
2) Priority 2 of the Client:

a) The Academic Planner Information Page: The client wanted major wise course requirements to be displayed for the user's knowledge. We successfully implemented this by connecting it to the server.
b) Checking off courses: The client imagined users to check the courses that they have already completed to see how many courses they still have left to complete.
c) Displaying the overall progress of the user's academic plans. We have the page "Graduation Requirements" that shows the prototype the client desired. We began working on "Class Information" but were not able to complete it with this timeline.

Known problems

- Could not figure out in time how to obtain credentials which could apply to all users rather than just Lena. For now, we have OAuth.
- Issues with server prevent all sections relying on server data from working correctly on certain machines. This problem is unsolved.
- Distribution requirement progress bar in "Graduation Requirements" does not immediately update as the user checks and unchecks each requirement.

Changes from the Original Plan

We did not deviate too significantly from the client's original idea. There were only very small User Interface changes that we made.

- The client pictured drop-down feature for Beyond the Classroom Page but we pitched making cards because that let us express the information more comprehensively
- The client wanted an overall progress page, but we instead made the Schedule page to show an overall picture of the semester's deadlines and the graduation requirements to show academic progress. We could not figure out how to quantify overall progress
- We decided against implementing a daily schedule feature because we agreed that most students do not have trouble remembering when their regularly scheduled classes are.
- Our customer was unsure of whether the Academic Planning section would be useful, but we insisted that it would be helpful to have all MHC-related information in one application.