Developer Manual

Student Code Online Review and Evaluation (2.0)

Student Code Online Review and Evaluation (2.0) or S.C.O.R.E (2.0) is a platform for creating and submitting programming assignments.

Team Members

Dorothy Ammons - dammons2022@my.fit.edu
Patrick Kelly - pkelly2022@my.fit.edu
Shamik Bera - sbera2022@my.fit.edu
Rak Alsharif - ralsharif2021@my.fit.edu

Faculty Advisor and Client

Raghuveer Mohan

April 19th, 2026

Table of Contents

Introduction

1.1 About S.C.O.R.E (2.0)

Student Code Online Review and Evaluation (2.0) or S.C.O.R.E (2.0) is a platform for creating and submitting programming assignments. Professors are able to create classes, assignments, rubrics, test cases and rosters. Additionally, they may view grades, submissions, AI usage scores and similarity scores. Students may use the application to view and make submissions to their assignments, receiving automatic grades and feedback through their submission output.

1.2 Requirements

In order to develop S.C.O.R.E (2.0) as expected, the following are required:

- Internet access
- Windows or Linux operating system
- Google Chrome installed with the desired S.C.O.R.E (2.0) login email
- S.C.O.R.E (2.0) executable
- GitHub
- Projectscore2.0@gmail.com login
- npm
- Python
- PyInstaller
- scikit-learn
- PyTorch

1.3 Google OAuth Data Collection

Our platform uses Google OAuth as a verification and identification tool. We store email addresses to remember and identify users. We do not collect or store any other personal data. We do not sell user data to third parties.

File Organization

2.1 GitHub Repository

The GitHub repository is connected to the projectscore20 GitHub account. The following command will allow you to fork your own copy of the repository.

git clone https://github.com/projectscore20/Automated-Grading-System-SCORE-2.0.git

You may then navigate to the directory.

```
cd Automated-Grading-System-SCORE-2.0
```

You now have the S.C.O.R.E (2.0) files locally.

Additionally, the webpage that has the loading page is under the projectscore20.github.io repository.

2.2 File Structure

The main directories that you would need are the

- Backend
    - API endpoints
    - Auto test logic
    - Similarity detection logic
- ai_detection
    - AI detectors
    - AI code detection logic
- autoTest
    - Auto feedback and grading logic
    - Auto test logic
- web_app
    - Vite configuration
    - Src files
        - Routes
        - All frontend files
- dist
    - Frontend build

2.3 Backend Files

Your main backend file is going to be app.py. This file contains just about every endpoint for every route. It controls Flask, threads, and database connections.

Your auto test files will be in the autoTest folder. This includes auto_feedback_object.py, which compares output, applies grades and stores results. The other important file is auto_test_object.py. This file compiles the code, runs the code, captures errors and runtime, and stores output. Supported languages are Python, Go, C, C++, Haskell, Rust, and Java.

Any changes you make to these files should be copied to the same autoTest file in the original directory.

Lastly, your similarity score logic is in the cops.py file. This is the file that requires scikit-learn and compares similarities between submissions.

2.4 Frontend Files

Your frontend files will be in the web_app folder. This folder contains your index.html, vite configuration, and src folder. In the src folder, you will see your components, assets, routes and a collection of jsx and css files. These are what create the visuals and interactions of the web application.

Database

3.1 Firebase

Firebase is the cloud database we use for the application. To reach the console, go to console.firebase.google.com, login with the projectscore2.0 email and click on Score20. The ID is score20-a5ae0. On the left panel you can navigate to Firestore. This is where you will see all stored files related to users, classes, and assignments. Right below it will be the Storage shortcut. Here you will see larger files stored including, descriptions and submissions.

Firestore allows for 20k writes, 50k reads, and 20k deletes per day on the free tier. Storage is through Google Cloud Firestore which allows for up to 1 GiB in total free.

3.2 Google Cloud Service Accounts

To reach the Google Cloud Console navigate to cloud.google.com and login with the projectscore2.0 email. Next, click the “Console” button in the top right. You should see the Score20 console with the same ID as the Firestore console.

To reach the service accounts, navigate to APIs and Services, then on the left click Credentials. Here you will see your browser key, OAuth Client ID, and service accounts. The OAuth Client ID is what allows the application to use Google OAuth for logins.

Building the Application

4.1 React Build Process

To begin building your application for use, first create the dist file. In your terminal, go to your project folder. Then, proceed into the web_app folder. Here you will run the following command:

```
npm run build
```

Once this process is complete, open the web_app folder in your file management application. Copy the new folder titled “dist” into your main directory and Backend folders. You will have to do this every time you make changes to any of your frontend files.

4.2 PyInstaller

To wrap your files into an executable, use PyInstaller. From your main directory in your terminal run:

```
python -m PyInstaller --onefile –noconsole --add-data "dist;dist" --add-data "Backend/score20-a5ae0-633d295581e6.json;Backend" --add-data "ai_detection;ai_detection" --hidden-import=cops --exclude-module torch.utils.tensorboard --exclude-module tensorboard Backend/app.py
```

Alternatively, on Linux machines run:

```
python -m PyInstaller --onefile --add-data "dist:dist" --add-data "Backend/score20-a5ae0-633d295581e6.json:Backend" --add-data "ai_detection:ai_detection" --hidden-import=cops --exclude-module torch.utils.tensorboard --exclude-module tensorboard Backend/app.py
```

With this command, you may have to use venv. If you receive errors, from your project root try running:

```
python3 -m venv venv
source venv/bin/activate
pip install --upgrade pip
pip install pyinstaller
```

Then continue with the PyInstaller command.

Once the build is complete, your application will be located in the dist folder in your main directory.