Event Horizon Calendar: Deployment Document

Jonathan Lavoie, Mason Cacheino, Nooh Alavi, Rahif Haffeez, Shawn Xiao

Overview:

This document is a guide that outlines the steps on how to package and deploy the Event Horizon Calendar for both the build and release versions. The application consists of a frontend made of HTML, CSS, and Javascript, a backend made from Python, with Flask to communicate between the two. The packaging process uses pre-made scripts that are provided in the GitHub repository.

Prerequisites:

- Python 3.8+ installed
- PyInstaller
- Virtual environment to isolate project dependencies (optional)

Deploying for the Build Version:

1. Go the the GitHub repository
2. Fork the repository, after forking you can clone the repository to your machine.
3. Set up Python Virtual Environment (optional)
4. Install Python Dependencies into the environment
5. Make changes that improve or add onto the application.
6. Start the application using main.py, after you can open the link to the Flask server (http://127.0.0.1:5000/). Test the functionality of the application here and ensure there are no bugs.
7. After testing of the application is finished, you can push those changes to your GitHub fork.
8. After pushing changes, you can make a pull request to add your changes to the original repository. If there are any changes to the original repository since you've forked, pull those changes into your fork.
9. After your pull request is made, a member will review it later on.

Deploying for the Release Version:

10. Go the the GitHub repository

11. Fork the repository, after forking you can clone the repository to your machine.
12. Set up Python Virtual Environment (optional)
13. Install Python Dependencies into the environment
14. Make changes that improve or add onto the application.
15. Start the application using main.py, after you can open the link to the Flask server (http://127.0.0.1:5000/). Test the functionality of the application here and ensure there are no bugs.
16. After testing of the application is finished, ensure that the code is production ready, this includes:
    - Testing: All features work as expected
    - Fixing Bugs: All bugs have been fixed
    - Code Cleanliness: Unnecessary code and ensure that your code is well documented.
    - Ensure that the versioning for your project is correct according to semantic versioning.
17. Using the provided build scripts in the repository, use them to package the application with PyInstaller.
18. Create a zip file with the project files and a simple batch/shell file that can automatically run the project.
19. Create a release on GitHub and push the code to the main repository. Ensure that the versioning is correct according to semantic versioning.