Advanced Web Technologies

Miguel Bacharov

40416445

For the coursework, I have split the project into two pieces: front-end & back-end. I have been able to deploy both pieces onto a Heroku (free tier option) for an open access. Both of which can be found here:

Front-end live version – https://space-shadow.herokuapp.com/. The front-end is written in SvelteJs + (Sapper) code. This is a very powerful JavaScript library for creating front-end applications.

Back-end live version - https://space-shadow-api.herokuapp.com/. The back end is written in Python, with the use of *Flask* (microframework), amongst other plugins/extensions, such as: *Flask_Restful,* Flask_CORS, Flask_Mail, Firebase_Admin and APScheduler.

For a better project management, I have used 2 separate Git Repositories: 1 for the Client side using the (Svelte-Js + Sapper), and 1 for the backend side, which contains the python-flask instance.

Front-End GitHub Repo -> https://github.com/migbash-university/ad-web-tech-backend

Back-End GitHub Repo -> https://github.com/migbash-university/ad-web-tech-frontend

I have created a single repo with both the front end (client) and back-end (server) folders from the above 2 repos as an alternative for easier viewing of the project, located here ->

https://github.com/migbash-university/ad-web-tech

Differences from initial plan

Over the course of the project development for the coursework in Part #2 of the coursework, I have followed exclusively my initial plan of the implementation of my space-tracking website software for live launches and the ability for the use of live notifications. However, I had made some changes to the original plan due to time constraints and overall, keeping the project a bit simpler:

Differences

1) I have removed the feature of the original plan to "expand" a particular launch event card, as I believe that it is an unnecessary feature and it is not aesthetically pleasing to the user,
2) I did not have the time to implement the "in-browser" notifications for the user as this was a more complex task that I have believed it would be initially,
3) I have changed the colour layout of the website, to adjust it for better aesthetics,
4) I have removed the unnecessary "Partnerships" website component, as that was only done for aesthetic purposes,

5) I have changed up the layout for some of the pages for better aesthetic while developing the coursework as I have believed a particular feature / design would better suit that components UI/UX,
6) I have decided not to pursue the integration of a 3D globe on the front page showing the respective active launch sites around the globe, and instead have opted for a simple clock with a futuristic touch to it,

Enhancements

1) I have added the support for the use of the *email_link* password reset for a particular user,
2) I have added the support for creating dynamic emails through the use of Flask_Mail, and send out the emails upon different actions performed by a user on the website, such as: registering on the website, subscribing to the newsletter, launch alert notification email confirmation, are a few of such examples.
3) Added the ability for continuous checking with the backend (firebase) or upcoming launches with the use CronJobs with the APScheduler python module, and upon 15 minutes prior to a launch, trigger email notfications to subscribed users,
4) Improved the initial */explorer* page to */space_explorer* with a better design and have removed unnecessary and unrelated UI components (such as: the navigation bar and the footer) for a more immersive experience of the space explorer GUI.
5) I have implemented the use of a /settings page for the user to update their account information and have the ability to logout from the web-app, something (that is essentially a crucial feature for any app on the web today) that was not part of the initial designs and plan,
6) Included the ability to hover over pin markers on an earth visualization simulation to read more about a particular launch site,

Improvements & Future Changes

Over the development phase of my coursework, I have seen potential improvements that and changes that could benefit this project in the future, with the following changes:

… implementing the “in-browser” push notifications for better user engagement,

… implementing the initial design of the space “missions” tracking capabilities, with the proper outline of the galaxy view and discovery of the mission orbit or location in space (on a planet).

… add support for further support for more of planets information, as I currently only hold support for Earth, Mars, and Jupiter.

… add more satellites data on to the simulation for Earth and expand it into other planets / bodies, that have artificial satellites, such as: The Moon, Mars, Venus… etc.

… add support for mobile notifications that I was meant to implement with the use of Twilio, alongside the in-browser and email notification.

… add support for a live chat (like one in YouTube) for users that are following a live launch, and are watching the provide stream of the launch, but only for authenticated users,

… add support for an in-website forum chat for users to converse with the use of the AdvaSpace account to be able to participate in the chat,

… add more content for the /space_explorer/missions page with more in-depth descriptions of each missions and other interesting fact about them,

… expand the galaxy simulation page with further capabilities, such as: objects located on the Moon’s surface, objects located on the Martian surface,

… fix the many current UI/UX bugs found throughout the website, particularly that of the user side and the /space_explorer page,

… inform the user of *“inProgress”* changes happening on the website, such as: a loading icon when a user submits their email to subscribe to the newsletter,

… add better navigational buttons and links across the website, as there are some key missing buttons that can make a user stuck in a potential page, without a navigation bar, such as: */space_explorer/missions*, with no way to get back home, other than manually clicking the browser back button,

… allow for a user to remove a news article from their favourites from within their *accounts_settings* page, as this is currently a lacking feature,

… a user currently must reset their password link, through setting a new password in the settings page and then, they receive an email link with the actual password resetting action in a different url, provided by Firebase,

… allow for actual webcast live url for a target launch to be consistent with the target launch, as I am currently using a dummy video link for all of the upcoming launches, as there is no way of properly checking with an external API whether a launch will be livestreaming their event or not.

Challenges faced & achievements made

Through out this coursework, I have achieved many first-time milestones and achievements, as well as many challenges:

I have learned how to use the very and amazingly powerful web-gui library for rendering 3D graphics right onto the browser, known as three.js. It was a difficult read of its documentation, but I am proud of the result.

I have learned how to use, to a very high degree of confidence, the library SpaceKit.js (a wrapper of the Three.js library, used solely for the development of interactive 3D space objects right in the users browser) , which has enabled me to develop my visualizations of space, individual celestial bodies and the visualization of orbiting satellites (as in the example with the planet Earth).

The hardest challenge in their project, was gathering the data and streamlining it with my custom Flask-RESTful implementation. I have struggled immensely with the aggregation of space data manually, as I was unable to find for a very long time the necessary open-APIs on different sorts of data, and had to in the end revert to create my own API, for key data, such as: space missions data, Earth Satellites and their unique descriptions and other information’s.

Learning how to calculate the orbit of a satellite through the use of its TLE data supplied from different open API sources and using the *Satellite.Js* library proved to be a slight challenge in figuring how to connect this library to that of *Spacekit.Js* for the actual visualization to work.

I have learnt how to properly manage a Python-Flask Backend Service with the use of the Flask-Restful API plugin for creating API Endpoints, and how to handle user sessions using both “cookies” and with the use of SvelteJs stores, for storing states for components and acting accordingly upon changes within the web-app.

I can now confidently create dynamic emails with the flask and customize them to my linking with no dependency on Firebase and other third-party services, and trigger them whenever I wish and / or as an action to a change.

Conclusion & Reflection

Overall, I have very much enjoyed the development of this website project. I believe that I have done more reading and expanding my capabilities of understanding different physics simulation-based libraries over the course of the development of this project, rather than spending time doing coding.

As the website stands, I believe it has a lot of features and components that I already have in my mind ready to be implemented for better user engagement and an overall better experience.

To attain the un-met challenges that I have posed to myself on my initial coursework preparation, I would begin with the implementation of the “Notification” pop-up enhancement to include the “Browser” & “Mobile” extensions. And I would then move onto the development and enhancement of the user on-boarding as I believed that I have not done it to the best of my ability. And finally **adding a lot more content**, in terms of planet insight info’s, mission’s info’s, more news articles, etc. More information can be found on my *ScreenCast Video*, supplied alongside with my submission.

For UI/UX changes done to the project throughout the development of it, can be found on my Figma Link -> https://www.figma.com/file/8nIo4axFPFyFT0YntF00wC/University-Projects?node-id=408%3A1061.