Ambioris Lora

Mobile App Programming

Fall 2021

Professor Levkowitz

Ikea Video App

Origin of Idea

The idea of this project came from the fact that me and my girlfriend frequently purchase things from Ikea and we were discussing ideas for an app. When I was doing the Survey assignment, she suggested that I should have an app that has a bunch of videos for every Ikea furniture all in one place. We both agreed that it is odd that Ikea themselves do not have videos of them building each furniture. Compound the need with the uniqueness of this app since I have never heard of anyone making an Ikea video app and this became my project.

First Step and Problems

The first step when it comes to making my Ikea Video App was to figure out how I am going to secure information for my app. There were multiple problems I had to account for while focusing on this aspect. The first one was the question “Should this information be stored locally within the app files?” I personally dislike this approach because it does not allow for modularity of your information, and it also results in a very “manual” way of doing things which I also dislike. This turned me to the API (Application programming interface) approach where I would host an API on the internet for me to access and modify information (The CRUD approach). This came with a couple of questions, for example, how would host both the scripts for the API and which database manager should I use? Due to the constraints and complexity of databases and scripts I choose something I was more familiar with and that is PythonAnywhere. This website service provides both a MySQL server, a constant uptime of scripts and a web domain hosting service. So, this solved a lot of problems at once. The website itself is free which is also a positive boon.

Second Step and Problems

After I figured out the “general framework” that I needed I first began working on the API. The reason as to why I began working on this first was for me to paginate my application I needed information. Pagination is the act of filling up your “page” with key information that follows a set formula for example if you had a blog post each post should have a name, text and date. In this case for my Ikea App, I needed to include Name, Designer, PDF, Video Links, and many measurements. I also did not want to hit a wall in development where I created the front end of the application but then I cannot work on anything else since I do not have a source of information to use on the app itself. Since PythonAnywhere is a Python only site it was clear that for my backend, I would simply use Python. There were a couple of issues that came up while developing this part. The first issue was how would I be able to publish this Api using python and how would I be able to generate CRUD actions on my database. Luckily for me there is the web framework “Flask.” Flask is a lightweight python web framework that allows me to have “routes” that handle several types of API requests. Once I got the routes setup, I was able to then visit these pages on the internet. The next portion was getting the JSON requests sent by me to arrive at the database on the PythonAnywhere SQL server. This was also made extremely easy with FLASK\_SQLALCHEMY. Which is a Python Module that allows the user to write to a database that they have the key information to. There was an issue with pip installing the module onto the PythonAnywhere server. This required me to use a virtual machine on the actual server to install the SQLALCHEMY module into a python3.8 environment. Once I installed the module, I was able to test several types of requests and routes.

Third Step and Problems

Once I got the API working, I had to create an online source of images for my app to constantly pull from. Hosting images on your local system tends to lack the modularity I like so I decided to host it online. I first searched online for an “image hosting website” and I found Cloudinary which had everything I needed to complete this step. You can upload images to the website, and it creates a personal folder of these images. You can upload either from your computer or using a web address. The web address aspect helped a lot because Ikea has web addresses for their images and so I just used those web addresses to upload those images. Cloudinary also assigns a personal web address to the images in your personalized folder, so this means that even if Ikea took down these pictures for some reason, you could still access them using a web address under your account which future proofs the project.

Fourth Step and Problems

It was at this point that I began working on the actual app itself. After experimenting with the base Android Development Kit, I realized that I particularly did not enjoy that form of development and thus began doing research on different approaches on mobile app development. I stumbled upon React Native which is a framework that allows you to create a mobile app for both iOS and android devices. Along with this boon you can also write it in JavaScript which I felt more comfortable in compared to java and XML. There were a couple of issues getting the android emulator working on my computer. The first one was my computer was not allowing virtualization which prevented me from running an emulator. This was a simple fix where I just had to enter BIOS mode and turn on hardware visualization. The next issue was allowing the run react-native app command to work. This required me to bypass the security policy every time I wanted to run the app.

Fifth Step and Problems

Once I got the whole setup working, I began working on the app. Since I previously created a Figma diagram that laid out the design aspect I did not have to think much about designing it on the go. The hardest part about this project was getting all the pieces together before the project even began. So once I got to this point all I had to do was create a couple of pages using react-native's navigation module and that was it. There were a couple of issues learning how exactly to paginate using the exact number of json objects in my database but when I found out about flat list in react native it became much easier. You can also pass parameters between pages using the react navigation module. Once I linked the Api to my app it just ended up working exactly how I wanted. The last big issue I ran into was handling YouTube videos in the app. The problem did not come from a programming aspect more from a NPM aspect. For some reason, for me to use this library I had to edit android grindle file to allow this module to be used in the app or else it would display a lot of errors. From this point on it just became a styling experiment to see if I wanted to change anything specific.

Conclusion

When I began working on this project, I had a general idea of what needed to be done when it comes to creating a full stack application, I just needed to figure out how to get it done. After doing research on the matter, I found out that my internship experience helped lay down the foundation necessary to create such an app without too much hair pulling. This app really made me appreciate mobile app development and I have hopes that in the near future, I can continue the road down learning more about this subset of software development.