Crypto DCA Calculator Mobile App

App structure

- **MVVM** was used to build the structure of the app. It uses DI to manage the services for the business work and also the view-models.
- **Supabase** was used for auth and DB services. For the authentication part, i will provide you with the credentials for an already created user(did not spend time creating the SignUp page). I will provide also some screenshots from the Supabase dashboard with the auth service and the DB.
- For the crypto prices **CoinGecko API** was used to fetch the actual prices(while testing the app please keep in mind that i used a free version of that api, so it won't give responses to requests with a timespan bigger than an year, you will see this mentioned later also with more details). The list with the name of the crypto coins is stored in the DB, so the app firstly fetches this list and with that it can make the request to CoinGecko API.
- App was tested only on Android, so if you will run it, please do it also only on Android, i did not have the time to test on iOS also(it should work without problems, but im not sure about the UI/UX).
- For input, charts, popups and some more i used **Syncfusion.** I used it to speed up the development process. Keep in mind that i used my own license(free license) to develop the app, so the GitHub repo won't get public.
- To version the app i used **GitHub** service. If required i can add you as a collaborator in the repo, i don't have a problem with that. If the case in a further discussion, i can share my screen if required.
- I did not build its own auth and business service API, i did not considered worth it for this MVP scenario. Since the project is focused on .NET MAUI.
- I did not had the time to implement WebSockets(btw, to really make a use out of WebSockets, the project description should be changed/add to something like, "add a live data chart with bitcoin price").
- Related to the 4 time of example from the doc i get with the instructions, i created 3 separate Views, *DcaSimulatorView* which corresponds to the *Basic* type of example, *DcaSimulatorIntermediateView* which corresponds to the *Intermediate* type of example, and *DcaSimulatorAdvancedView* which corresponds to the first *Advanced* example. To describe them in a few words: second and third one do contain the functionality of the first. Also the first one can be extracted as a separate component(future improvement). Only the third was cleaned up.

- Keep in mind that the app and the calculator were not fully tested(again, not enough time), so its not *bullet-proof*. I will provide a video on how to work with it and i can explain any part of code.

Credentials:
email: test@tokero.com
pass: 123456

For the selection of the date, to extract crypto prices from coin gecko, again, it wont work for for timespans bigger than an year, so please when testing choose dates only from the current 2025 year.