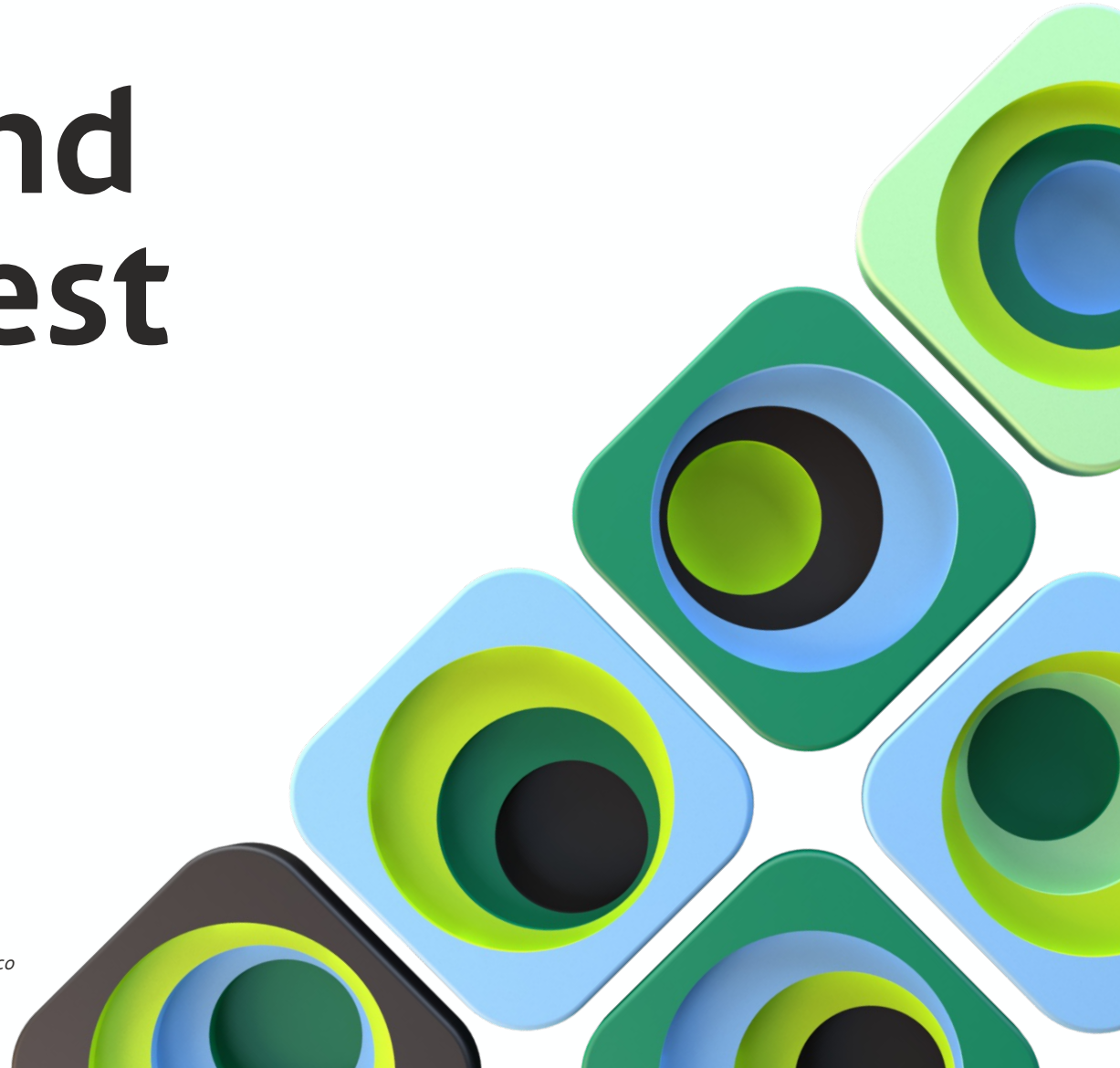
Mobile + Backend Development Test

Software Engineering

Tech team



Wompi

Last rev. 21/Jul/2025 *by, Javier Alejandro Perez javier.perez@wompi.com, Laura Ramirez laura.ramirez@wompi.co*

About the test

In this test you must create a **Credit card payment checkout**

It is the onboarding where we obtain customer payment data to make the transaction of specific product and show the result of the process[1].

This App follows 7-step screen business process:

→ **1. Splash Screen:** Create as your own consideration

→ **2. Home of Products:** You must create a UI to show the product from your store (with information and value), that will be bought by the customer.

→ **3. Select Product:** The customer will select or add 1 o N items.

→ **4. Checkout:** Show to the customer the payment type button "*Pay with credit card*". That opens a Backdrop requesting Credit Card info.

→ **5. Credit Card info:** Customer will insert Credit Card data and must be validated. MasterCard and VISA number card detection logos is a plus. *(Remember, credit card data must be fake but must follows the structure of credit cards)*

→ **6. Payment Summary:** Next show the summary payment with payment button in a backdrop component[2]. Then make backend API call with happy and unhappy paths for the payment (regarding if the payment data is complete, in case of unhappy show a toast error). *(See details about backend API on 4th slide)*

→ **7. Final transaction status:** Finally show the result of the transaction made and go back to Home of Products.

Your app must be resilient, the App save state, if necessary, crashes are not allowed.
Attention to detail is key.

Wompi

References

1. https://soporte.wompi.co/hc/es-419/articles/360057781394--C%C3%B3mo-pagar-con-tarjetas-a-trav%C3%A9s-de-Wompi-
2. https://m2.material.io/components/backdrop

About the test

Minimum requirements for Mobile App

1. You must implement a mobile app built in **React Native (use last stable version)**. NO other frameworks/tools/libraries or native development are allowed.

2. Create a Front that will manage multiple screen sizes with anchors. Making your App at minimum responsive. (Minimum screen size reference: iPhone SE (2020) 1334 x 750 pixels). UI interaction will work properly and must fit into UI boundaries.

3. Use of Redux **is MANDATORY,** following Flux Architecture alignments as possible. Additionally, store the payment transaction data securely in the *application state* or *localStorage. (Encrypted!)*

4. UX Design could be created as your own consideration.

5. Use Styles of your preference.

6. **Unit tests is MANDATORY** for the competition of the tests, more than 80% of the coverage. (*Add the results in README.md and create them with Jest*)

7. Compile your mobile app in your local computer (either Desktop or Mac). Please upload the **.apk (Android)** and/or **.ipa (iOS is optional)** in your repository ready to use.

Considerations

✅ We recommend create branches and pull requests by feature.
⚠️ Include a README file with all necessary instructions for running the app and tests.
⚠️ You must push your code to Github as a public repository only!
⚠️ Please do NOT use words like “Wompi” (company name), in your repository, and do not share your solution with other developers or candidates.

Wompi

About the test

Minimum requirements for Backend API

1. You must implement a backend API **TypeScript + Nest.js** ***(use last stable version)***. NO other frameworks or languages are allowed.

2. When user click payment button:
    1. Create a transaction in PENDING in your backend API and obtain a transaction number.
    2. Call **Wompi Payment API** to complete the payment. *(See details on next slide)*
    3. Once payment is completed or failed:
        1. Update the transaction with the transaction result in your Backend
        2. Assign the product that will be delivered to the customer
        3. Update the stock of your product.

3. **Unit tests is MANDATORY** for the competition of the tests, more than 80% of the coverage. (*Add the results in README.md and create them with Jest*)

4. **Publishing the app using any Cloud provider is OPTIONAL,** if not please create a dockerfile ready to use. *(Cloud deployment works better and easier to integrate within the API layer on the mobile App, connecting directly through the backend API)*.
    1. We recommend AWS *(with the AWS free tier, AWS Lambda, ECS, EKS, CloudFront, S3, RDS or DynamoDB)*

Considerations

✅ We recommend create branches and pull requests by feature.
⚠️ Include a README file with all necessary instructions for running the app and tests.
⚠️ You must push your code to Github as a public repository only!
⚠️ Please do NOT use words like “Wompi” (company name), in your repository, and do not share your solution with other developers or candidates.

Wompi

About the test

Connection keys for Wompi Payment API (UAT)

1. Wompi API. For starters, you must read:
    1. https://docs.wompi.co/docs/colombia/inicio-rapido/
    2. https://docs.wompi.co/docs/colombia/ambientes-y-llaves/

2. Use the following information to use API Keys or to access to an account created for this test. Important note: This user could be used by any people doing the same test and your user session could be closed by different location accessing at the same time, try to integrate with API Keys. Is important always the tests must be on Sandbox mode. (Make sure you use the Sandbox environment. That means, there's NO need for any real money transactions, in the context of this test.)

    - Is important always the tests must be on **Sandbox mode**
    - **Sandbox mode** is in the main menu → Developers

    - Wompi User
        - URL: https://login.staging.wompi.dev/
        - User: procesoseleccionbackend@yopmail.com
        - Password: JobProcessWompi987*

    - API URLs
        - UAT_URL=https://api.co.uat.wompi.dev/v1
        - UAT_SANDBOX_URL=https://api-sandbox.co.uat.wompi.dev/v1

    - API Keys Sandbox
        - pub_stagtest_g2u0HQd3ZMh05hsSgTS2lUV8t3s4mOt7
        - prv_stagtest_5i0ZGlGiFcDQifYsXxvsny7Y37tKqFWg
        - stagtest_events_2PDUmhMywUkvb1LvxYnayFbmofT7w39N
        - stagtest_integrity_nAIBuqayW70XpUqJS4qf4STYiISd89Fp

Wompi

About the test

Deliverables

1. Completion of minimum requirements
2. Link of GitHub repository with README.md updated.
3. App and Backend API compiled successfully and uploaded in the repository.

We evaluate

1. **[5 points]** Readme.md complete, Images that renders fast, avoiding UI/UX Out-boundaries and app without crashes
2. **[35 points]** Mobile App with the onboarding process for Credit card payment checkout.
3. **[30 points]** Backend API, with payment management working correctly.
4. **[30 points]** Unit test and coverage

5. **[15 Bonus points]** CSS skills.
6. **[20 Bonus points]** Clean code and/or Hexagonal Architecture pattern applied.
7. **[25 Bonus points]** Backend API deployed on Cloud.

Considerations

✅ To complete the test, you need at least 100 points.
✅ If you obtain the minimum points will continue with the interview step.
⚠️ If it is detected that the repository does not have progress or commits, or that it is like the solution of another candidate, the test will be automatically voided for fraud.

Wompi

References
1. *https://observatory.mozilla.org/*


Wompi