IMPAKT Group

Project: **NBP Currency Analysis System**

Members:
Damian Opolski (Scrum Master, Developer) - 202280
Hubert Cywka (DevOps) - 240645
Łukasz Kałużny (Developer) - 240695
Kacper Kornacki (Developer) - 221022
Alan Licata (Tester) - 241223

Acceptance tests report

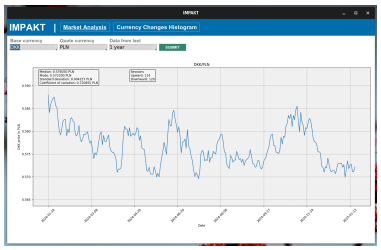
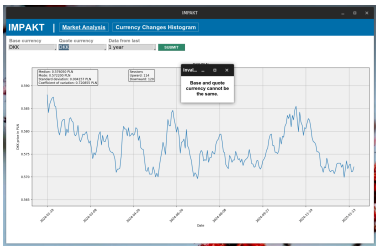
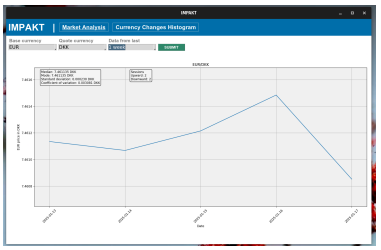
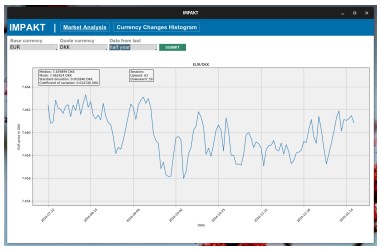
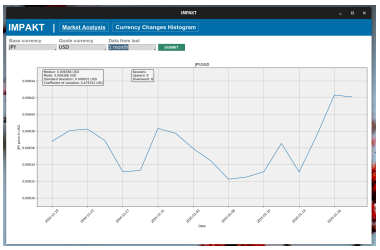
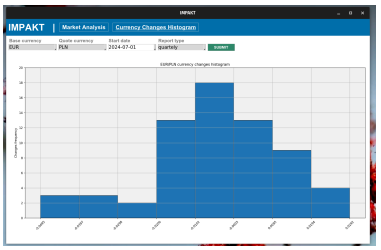
The acceptance tests were conducted to evaluate the functionality, usability, and display aspects of the application that shows exchange rate charts between two selected currencies over a given period of time. The focus was on ensuring that charts displayed correctly, data nodes were not obstructing important information, and error handling was appropriate. Initially, the tests ran resulted in errors regarding the way some of the UI elements were displayed, but they were fixed by the latest updates, and all the tests were passed successfully.

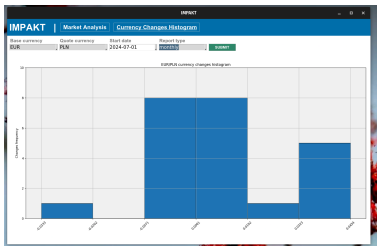
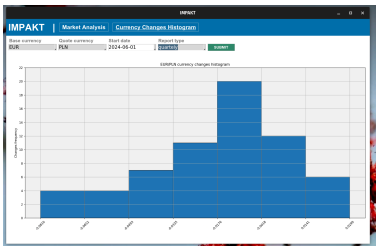
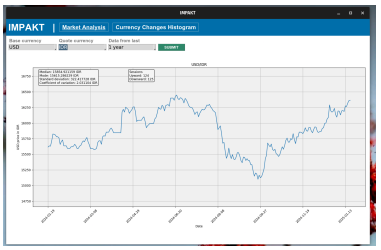
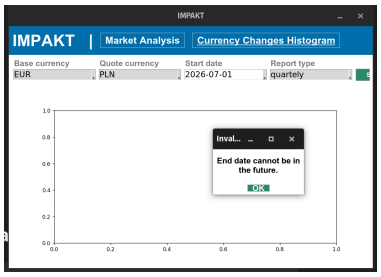
Test Results

Test No.	Test Description	Result	Comments
1	Tested currency exchange rate calculation for pair **DKK/PLN** over the last 1 year period.	Succeeded	
2	Tested currency exchange rate calculation for pair **DKK/DKK** over the last 1 year period.	Succeeded	
3	Tested currency exchange rate calculation for pair **EUR/DKK** over the last 1 week period.	Succeeded	
4	Tested currency exchange rate calculation for pair **EUR/DKK** over the last half-year period.	Succeeded	
5	Tested currency exchange rate calculation for pair **JPY/USD** over the last 1 month period.	Succeeded	

6	Tested currency exchange rate calculation for pair **EUR/PLN** over custom period (2024-07-01) with "quarterly" report.	Succeeded	
7	Tested currency exchange rate calculation for pair **EUR/PLN** over custom period (2024-07-01) with "monthly" report.	Succeeded	
8	Tested currency exchange rate calculation for pair **EUR/PLN** over custom period (2024-06-01) with "quarterly" report.	Succeeded	
9	Tested currency exchange rate calculation for pair **USD/IDR** over the last 1 year period.	Succeeded	
10	Tested currency exchange rate calculation for pair **EUR/PLN** over the wrong period of time, starting from the future date, with “quarterly” report.	Succeeded	Program showed the error message as expected.

Screenshots:

Test no 1: 	Test no 2: 	Test no 3: 
Test no 4: 	Test no 5: 	Test no 6: 

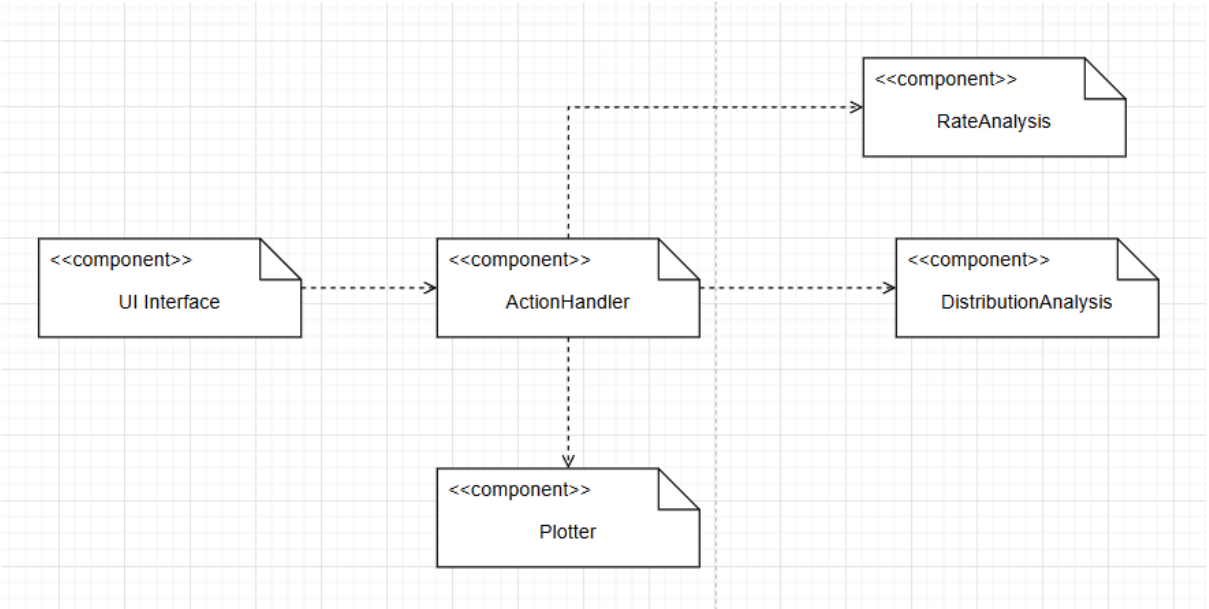
Test no 7: 	Test no 8: 	Test no 9: 
Test no 10: 		

Architecture description

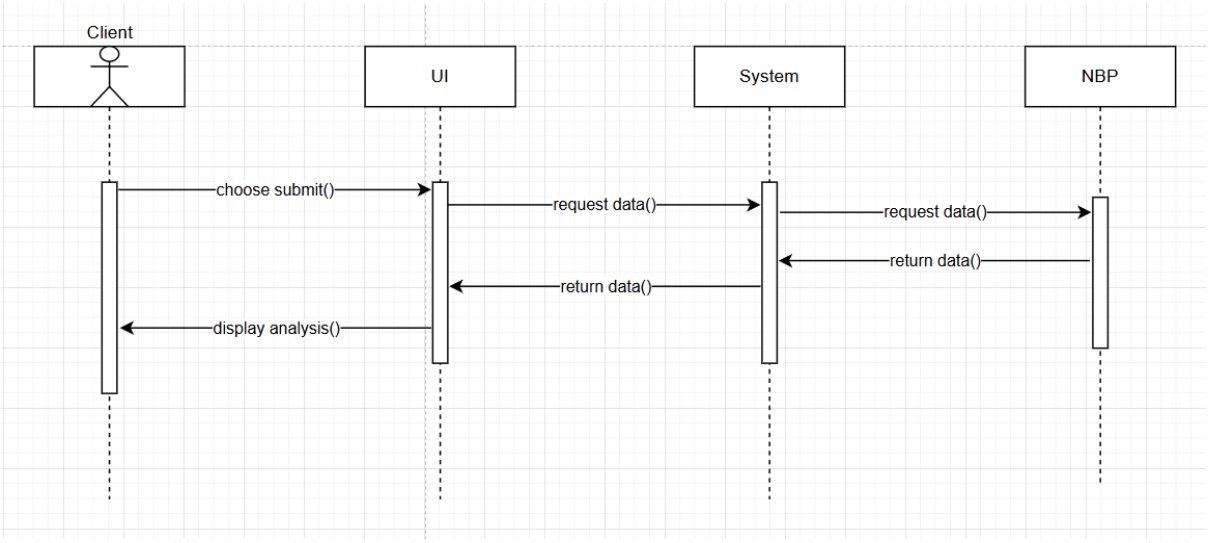
- Technologies:
  - Programming Language: Python is the primary programming language used for the implementation of the application's logic, including data retrieval from the NBP API, statistical calculations, GUI logic, and unit testing.
  - Python Libraries:
    - requests: Used for making HTTP requests to interact with the NBP API and retrieve exchange rate data.
    - numpy: Leveraged for numerical computations and efficient handling of data arrays in statistical analysis.
    - tkinter: The standard Python GUI toolkit employed to create the application's graphical user interface.
    - matplotlib: Used to generate visualizations, particularly histograms, for displaying the distribution of currency changes.
    - statistics: Python's built-in module is used for common statistical calculations such as median, mode, standard deviation, and mean.
    - unittest: Python's framework for writing and running unit tests to ensure the reliability of the system components.
    - parameterized: A library used to parameterize unit tests, facilitating the testing of multiple scenarios using a single test definition.
    - Tools: The system primarily relies on Python and its associated libraries. No additional tooling is directly incorporated within the software.

- Environment:
    - Target Platform: The application is compiled into a standalone executable file (.exe) and designed to be run directly on client machines using Windows operating systems. This is not a web application or a server-based service.
    - Dependencies: The application relies on a Python runtime environment, including the dependencies listed above. The compilation process should package these dependencies within the executable.
    - Serverless Deployment: The system does not require a server-side infrastructure; the application fetches data directly from the NBP API on the user's machine.
- Delivery:
    - Process:
        - Coding and Testing: Python code is developed and tested to ensure correct functionality.
        - .exe Compilation: The Python code, along with its dependencies, are packaged into a single executable file (.exe) using a tool like PyInstaller or auto-py-to-exe.
        - GitHub Actions: GitHub Actions is implemented to automate the build and .exe generation. The GitHub Actions workflow is configured to run automatically upon specific triggers such as pushes or merges to the main branch, ensuring consistent and repeatable builds.
        - Distribution: The resulting .exe is made available for users to download and execute directly on their Windows-based systems.
    - Automation: The entire deployment process, including building the executable and making it available, is automated using GitHub Actions.
    - Distribution method: The generated .exe file will be distributed outside of a dedicated app store, most likely via direct download links or hosted in a Git repository.

Components diagram

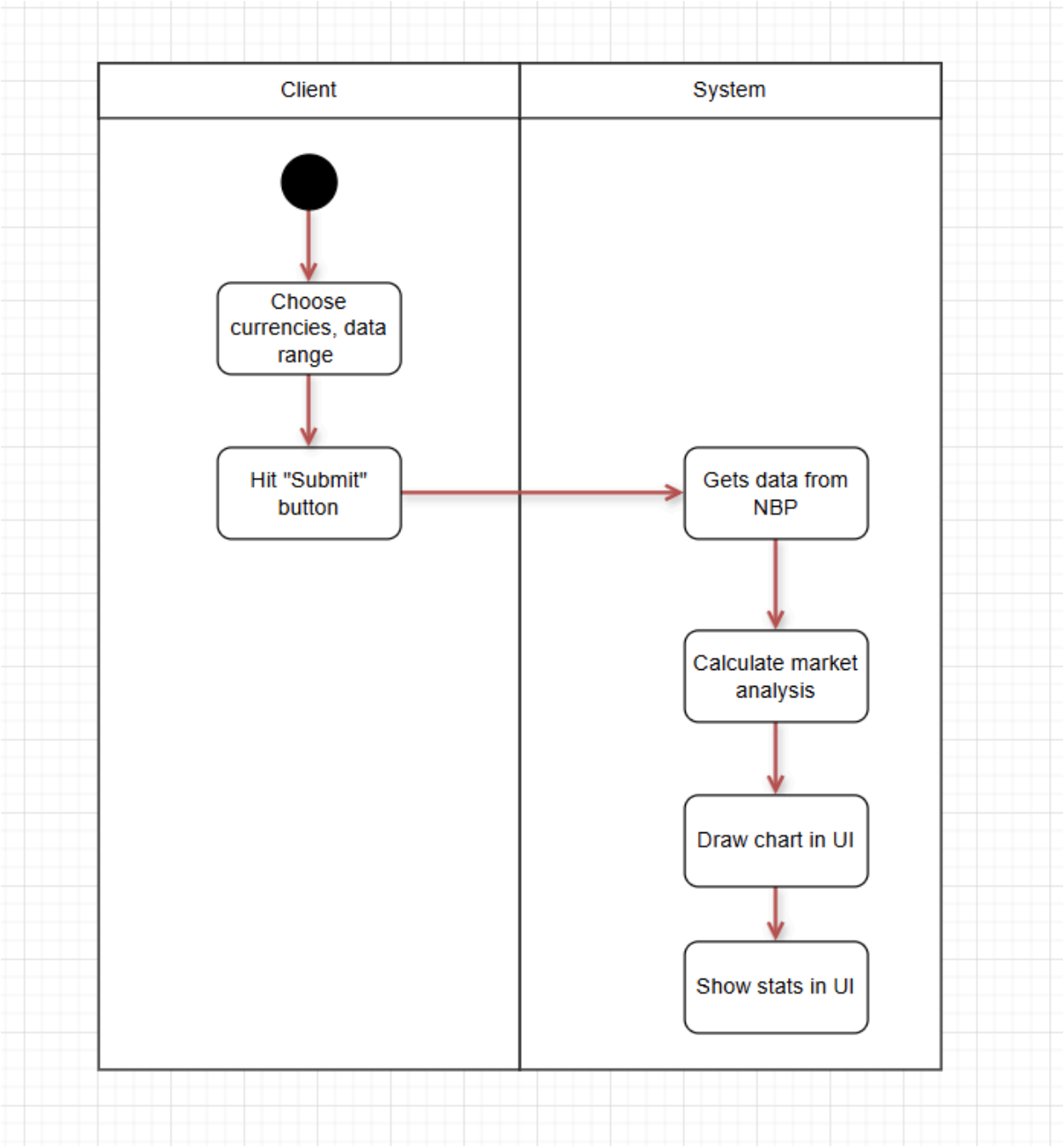


Sequence diagram



Activity diagram

Market Analysis



Currency Changes Histogram