Stark-Tower Reimburse

Project 1 Reimbursement system

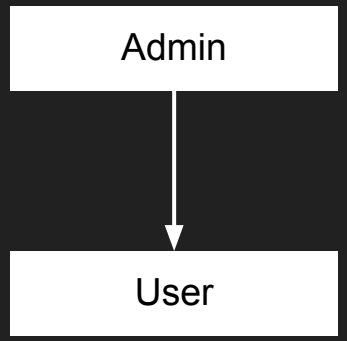
General Design choices

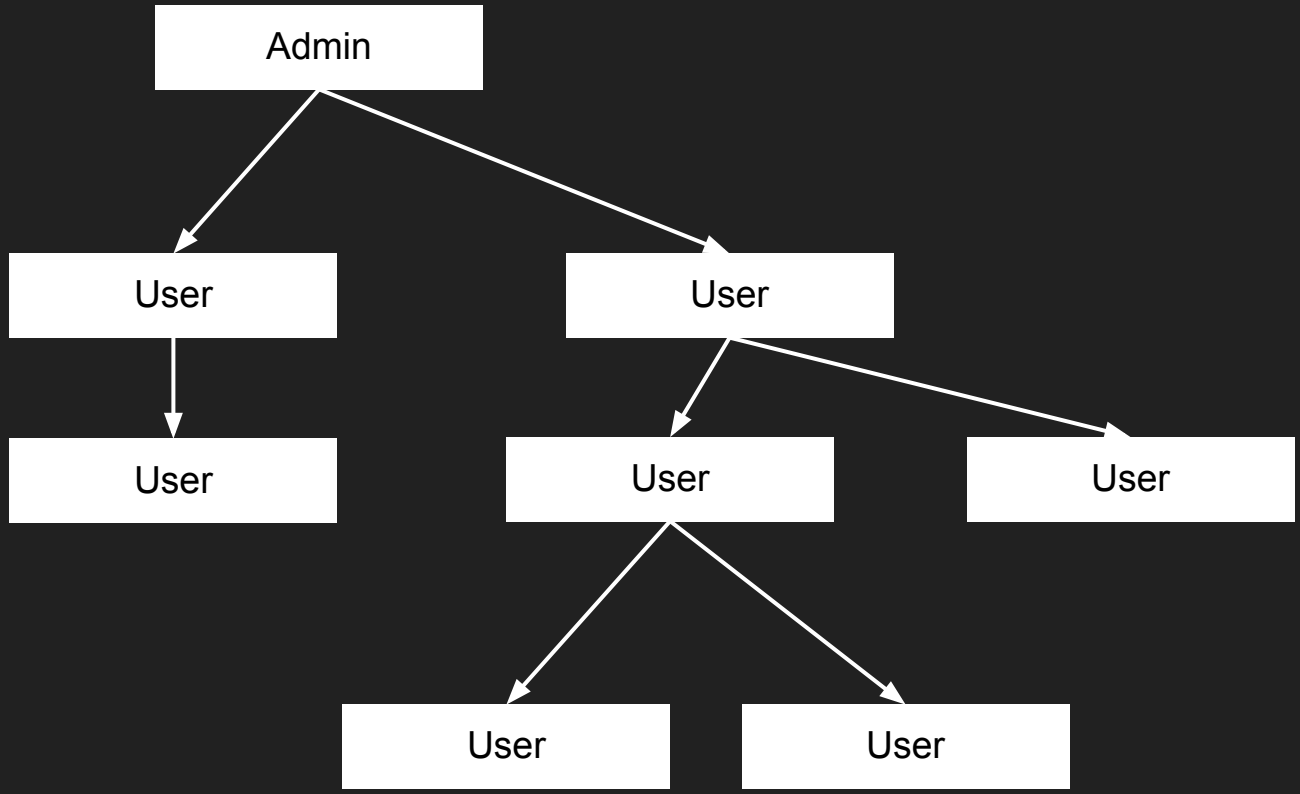
- The Server controls all correct data
  - All data manipulation is done on the server
    - approve, deny, create new reimbursement…
  - All data process is done on the server
    - The server does all filter
- The Frontend is exclusively for show data
  - No data manipulation is done on the front end
  - It can only display information the backend sends
  - Only a handful of buttons sends commands to the backend
- Why this design?
  - The front end can be forced on displaying and nothing else
  - If the format of the data changes, the front end doesn’t break
  - Statistics can be done any way and not change how the front end looks
    - Unless something is deprecated

General Design choices

- Structure
  - All Accounts have the following structure
    - Each Account has 1 Manager
    - Only Manager your assigned employees
  - If you have no Manager
    - You are an admin
    - You can manager ALL employees
  - Why?
    - A manager could make a request to their supervisor
      - This can be disabled if all managers are admins
    - You can easily change the structure as you need it
      - 1 admin
      - Branching tree after this

Structure

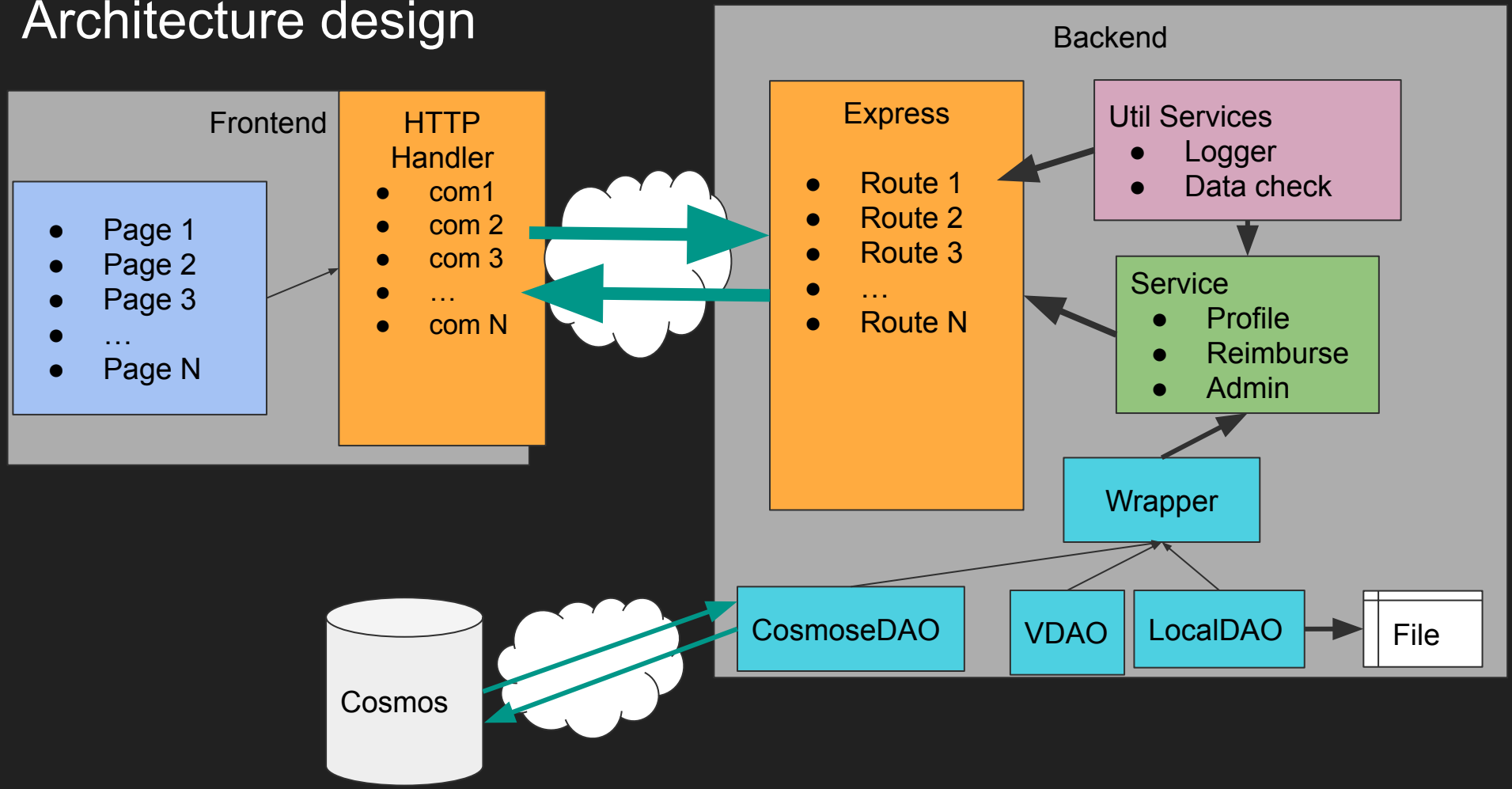




In app

- Statistic page
  - Frontend: request some records
  - Backend:
    - Grabs all reimbursements
    - Sums a total for each employee
    - Create a string array => ‘${FirstName} {LastName} : {TotalSum}’
    - Return this Array
  - Frontend: Display this string array
- If changed*
  - If the server sorts, the front end do NOTHING different
  - Multiple sorting behaviors can be used
    - All array are displayed in a row
  - The front end does not change
  - This is how Request View page works

Architecture design



Style Architecture design

- NO STYLING IN TOP LEVEL COMPONENTS!!!!
  - Created several basic component with specific parameters
    - Created some Enums
      - Text: Title, Header, General
      - Button: General, Close, Admin Tool
      - Input text
      - …
  - From here the style for the component comes from a function
    - `StyleComponent(type) >>`
      - `{ color:GetColor(type), flex:1, padding:4, …}`
    - `<Component style={ StyleComponent(type) } … />`
  - This supports full app styling
    - All styling is done in a single StyleSheet configuration file
    - Different theme are *'swappable'*
    - All color scheme are consistent
  - The only 'styling' in component is positioning
    - Only done by using View