User's main page

The main page of the application, displaying the user profile, workout statistics, and workout categories.

Props:
- user: {object} – user information (avatar, name) *(required)*
- statistics: {array} – array of data for the statistics chart *(optional)*
- categories: {array} – list of workout categories (title, image) *(optional)*
- onProfileClick: {function} – function triggered when the user clicks the profile icon *(required)*
- onCategorySelect: {function} – function triggered when the user selects a category *(required)*
- onStatisticsClick: {function} – function triggered when the user clicks on statistics *(required)*

States:

Behavior:
- Clicking the profile icon calls onProfileClick(), navigating to the "Profile information page".
- Clicking on a category calls onCategorySelect(category), navigating to the "Start Training" page.
- Clicking "See full statistics" calls onStatisticsClick(), navigating to the "Statistics page user".

Composition:
- **Parent components:** App
- **Child components:** ProfileIcon, StatisticsChart, CategoryGrid

Variations:

ProfileIcon

The user profile icon.

Props:
- userImage: {string} – URL of the user avatar *(required)*
- onClick: {function} – function triggered when the user clicks the profile icon *(required)*

States:

Behavior:
- Clicking the profile icon calls onProfileClick(), navigating to the "Profile information page".
- Placeholder icon if no image is provided

Composition:
- **Parent components:** User's Main page
- **Child components:** None

Variations:

StatisticsChart

A chart displaying the user's workout statistics over the week.

Props:
- data: {array} – an array containing workout statistics for the week *(optional, default: [])*
- onClick: {function} – function triggered when the user clicks the chart *(required)*

States:

Behavior:
- Displays user statistics in a bar chart format.
- Absence of the statistics if the user is new.
- Clicking "See full statistics" calls onClick(), navigating to the "Statistics page user".

Composition:
- **Parent components:** User's Main page
- **Child components:** None

Variations:

CategoryGrid

A grid displaying workout category cards.

Props:
- categories: {array} – list of categories with titles and images *(required)*
- onCategorySelect: {function} – function triggered when a category is selected *(required)*

States:

Behavior:
- Displays a list of workout categories.
- Clicking on a category calls onCategorySelect(category), navigating to the "Start Training" page.

Composition:
- **Parent components:** MainPage
- **Child components:** CategoryCard

Variations:

CategoryCard

A single workout category card.

Props:
- title: {string} – category title *(required)*
- image: {string} – category image *(required)*
- onClick: {function} – function triggered when the card is clicked *(required)*

States:

Behavior:
- Clicking the card calls onClick(title), navigating to the "Start Training" page for that category

Composition:
- **Parent components:** CategoryGrid
- **Child components:** None

Variations:

Profile information page

The user profile page displays personal information and allows editing. The "Apply Changes" button saves updates, and the back button navigates to the **User's Main Page.**

Props:
- user: {object} – user information including:
  - firstName: {string}
  - lastName: {string}
  - email: {string}
  - phone: {string}
  - address: {string}
  - city: {string}
  - membership: {string}
- onBack: {function} – function triggered when the user clicks the back button *(required)*
- onSaveChanges: {function} – function triggered when the user clicks "Apply Changes" *(required)*

States:
formData: {object} – stores the editable form fields *(updates when the user types in an input field)*

Behavior:
- Clicking the Back Button calls onBack(), navigating back to the User's Main Page.
- Clicking the Apply Changes Button calls onSaveChanges(formData), saving updated user information.
- Input fields allow the user to modify their details.

Composition:
- **Parent components:** App
- **Child components:** ProfileIcon, ProfileForm, Button

Variations:

ProfileIcon

Displays the user's profile avatar.

Props:
- userImage: {string} – URL of the user avatar *(optional, default: placeholder image)*

States:

Behavior:
- Default img if no image is provided

Composition:
- **Parent components:** ProfilePage
- **Child components:** None

Variations:

ProfileForm

A form containing the user's editable profile fields.

Props:
- formData: {object} – contains user information fields *(required)*
- onChange: {function} – function triggered when a user edits a field *(required)*

States:
editedData: {object} – stores modified user input *(updates dynamically)*

Behavior:
- Allows the user to edit name, email, phone, address, city, membership type, and password.
- Calls onChange(updatedField, value) when the user types in an input field.

Composition:
- **Parent components:** ProfilePage
- **Child components:** InputField

Variations:

InputField

A reusable input field component for text, password, and other form inputs.

Props:
- label: {string} – label of the input field (required)
- type: {string} – input type (text, email, password, etc.) (required, default: "text")
- value: {string} – current value of the input field (required)
- onChange: {function} – function triggered when the user types (required)

States:
  value: {string} – Stores the current value of the input field (updates dynamically)

Behavior:
- Allows the user to edit name, email, phone, address, city, membership type, and password.
- Calls onChange(updatedField, value) prop function with the new value when the user types

Composition:
- **Parent components:** ProfileForm
- **Child components:** None

Variations:
- Type with validation (e.g., email format)
- Hidden type (for the password).

Button

A reusable button component.

Props:
- label: {string} – button text (required)
- onClick: {function} – function triggered when the button is clicked (required)
- variant: {string} – button style (primary, secondary, danger, etc.) (optional, default: "primary")

States:

Behavior:
- Calls onClick() when clicked.

Composition:
- **Parent components:** ProfilePage
- **Child components:** None

Variations:
- Primary, secondary, disabled designs.

Statistics page user

This page displays the user's workout statistics, including weekly and monthly graphs. Users can navigate between weeks and months using arrow buttons.

Props:
- weeklyData: {array} – array of weekly statistics (required)
- monthlyData: {array} – array of monthly statistics (required)
- onBack: {function} – function triggered when the user clicks the back button (required)
- onWeekChange: {function} – function triggered when navigating between weekly statistics (required)
- onMonthChange: {function} – function triggered when navigating between monthly statistics (required)

States:
currentWeek: {object} – stores the currently displayed week's data (updates when user navigates weeks)
currentMonth: {object} – stores the currently displayed month's data (updates when user navigates months)

Behavior:
- Clicking the Back Button calls onBack(), navigating back to the User's Main Page.
- Clicking the Left or Right Arrows on the weekly graph calls onWeekChange(direction), updating currentWeek.
- Clicking the Left or Right Arrows on the monthly graph calls onMonthChange(direction), updating currentMonth.

Composition:
- **Parent components:** App
- **Child components:** ProfileIcon, StatisticsGraph, NavigationArrows, PieChart

Variations:

StatisticsGraph

A reusable component that displays a bar chart for either weekly or monthly statistics.

Props:
- data: {object} – statistics data and data label (required)
- title: {string} – title of the graph (e.g., "This Week", "This Month") (required)
- totalHours: {number} – total hours for the period (required)
- onChange: {function} – function triggered when navigating between datasets (required)
- GraphType: {string} – what form will graph have

States:

Behavior:
- Displays a bar chart with user statistics.
- Clicking the Left or Right Arrows calls onChange(direction), navigating between periods.

Composition:
- **Parent components:** StatisticsPage
- **Child components:** NavigationArrows

Variations:
- Different graph types (line, area charts).

NavigationArrows

A component for navigating between different weeks or months.

Props:
- onPrevious: {function} – function triggered when clicking the left arrow (required)
- onNext: {function} – function triggered when clicking the right arrow (required)

States:

Behavior:
- Clicking the left arrow calls onPrevious(), navigating to the previous period.
- Clicking the right arrow calls onNext(), navigating to the next period.

Composition:
- **Parent components:** StatisticsGraph
- **Child components:** None

Variations:

Start training

This page allows users to track time spent on a specific workout category. Users can start a timer, pause it, manually input time and save the recorded time.

Props:
- category: {string} – name of the selected workout category (required)
- onBack: {function} – function triggered when the user clicks the back button (required)

States:
timeElapsed: {number} – the elapsed workout time in seconds (updates dynamically when the timer runs)
isRunning: {boolean} – whether the timer is currently running (toggles when starting/stopping the timer)
entryMode: {string} – whether the user is manually entering time or timer is used (updates when clicking "Enter manually" to "manual", default is "timer")
manualTime: {number} – the manually entered time in seconds (updates when user inputs manually)

Behavior:
- Clicking Start sets isRunning = true and starts the timer.
- Clicking Stop sets isRunning = false, pausing the timer.
- Clicking Save calls onSave(timeElapsed), storing the workout duration.
- Clicking Enter manually sets entryMode = 'manual', switching to manual input.
- Clicking Cancel when in manual mode returns to the timer.

Composition:
- **Parent components:** App
- **Child components:** TimerDisplay, TimerControls, ManualEntry

Variations:
- Timer entryMode to display timer.
- Manual entryMode to enter time manually.

TimerDisplay

Displays the workout duration in a digital clock format. Formats and displays time in MM:SS format.

Props:
- time: {number} – the current elapsed time in seconds (required)

States:

Behavior:
- Displays current training time

Composition:
- **Parent components:** WorkoutTimerPage
- **Child components:** None

TimerControls

Contains buttons to start, continue, cancel, stop, save, or manually enter workout time.

Props:
- isRunning: {boolean} – whether the timer is currently running (required)
- onStart: {function} – function triggered when the user starts the timer (required)
- onStop: {function} – function triggered when the user stops the timer (required)
- onSave: {function} – function triggered when the user saves the recorded time (required)
- onManualEntry: {function} – function triggered when switching to manual input mode (required)
- btnType: {string} – type of the button and what function will be triggered
- onCancel:{function} – function triggered when the user cancels time entry (required)

States:

Behavior:
- Clicking Start or Continue calls onStart(), beginning the timer.
- Clicking Stop calls onStop(), pausing the timer.
- Clicking Save calls onSave(), storing the recorded time.
- Clicking Enter manually calls onManualEntry(), opening manual input.

Composition:
- **Parent components:** WorkoutTimerPage
- **Child components:** Button

Variations:
- Button type (start, Enter manually, Stop, Cancel, Save, Continue).

Button

A reusable button component.

Props:
- label: {string} – button text (required)
- onClick: {function} – function triggered when the button is clicked (required)
- variant: {string} – button style (primary, secondary, danger, etc.) (optional, default: "primary")

States:

Behavior:
- Calls onClick() when clicked.

Composition:
- **Parent components:** TimerControls
- **Child components:** None

Variations:
- Primary, secondary, disabled designs.

ManualEntry

Allows the user to input workout time manually.

Props:
- onChange: {function} – function triggered when the user updates manual input (required)

States:
- minutes: {string} – the currently entered minutes (updates on user input)
- seconds: {string} – the currently entered seconds (updates on user input)

Behavior:
- User can input values into separate fields for minutes and seconds.
- As the user types in the minute or second fields, the component's internal 'minutes' and 'seconds' state is updated.

Composition:
- **Parent components:** WorkoutTimerPage
- **Child components:** InputField

Variations:

InputField

A reusable input field component for text, password, and other form inputs.

Props:
- label: {string} – label of the input field (required)
- type: {string} – input type (text, email, password, etc.) (required, default: "text")
- value: {string} – current value of the input field (required)
- onChange: {function} – function triggered when the user types (required)

States:
  value: {string} – Stores the current value of the input field (updates dynamically)

Behavior:
- Allows the user to edit name, email, phone, address, city, membership type, and password.
- Calls onChange(updatedField, value) prop function with the new value when the user types

Composition:
- **Parent components:** ManualEntry
- **Child components:** None

Variations:
- Type with validation (e.g., email format)
- Hidden type (for the password).

RegistrationPage

This page allows users to register by filling in their details (name, surname, email, access code, password). Upon successful registration, they can log in immediately or later.

Props:
- onRegister: {function} – function triggered when the user submits the registration form (required)
- onLoginRedirect: {function} – function triggered when the user clicks "Log In" (required)

States:
- formData: {object} – stores the user's input for name, surname, email, access code, and password (updates when the user types in fields)

Behavior:
- Clicking Sign up calls onRegister(formData), initiating registration.
- Clicking Log In redirects the user to the LoginPage (onLoginRedirect).

Composition:
- **Parent components:** App
- **Child components:** InputField, Button

Variations:

LoginPage

The login page where users enter their login credentials (login and password) to access the main page.

Props:
- onLogin: {function} – function triggered when the user clicks the login button (required)
- onForgotPasswordRedirect: {function} – function triggered when the user clicks the "Forgot password?" link (required)
- onSignUpRedirect: {function} – function triggered when the user clicks the "Sign up" link (required)

States:
loginData: {object} – stores the login credentials input (username, password) (updates dynamically when typed)
error: {string} – stores error message when the login fails (updates when wrong credentials are entered)

Behavior:
- Clicking Log In calls onLogin(loginData), logging the user in.
- Clicking Forgot password? redirects the user to the RecoverPasswordPage (onForgotPasswordRedirect).
- Clicking Sign up redirects to the RegistrationPage (onSignUpRedirect).

Composition:
- **Parent components:** App
- **Child components:** InputField, Button

Variations:

RecoverPasswordPage (Email Entry)

This page allows the user to enter their email to receive a recovery code for password reset.

Props:
- onSendCode: {function} – function triggered when the user submits the email (required)

States:
email: {string} – stores the user's entered email (updates dynamically when typed)

Behavior:
- Clicking Send code calls onSendCode(email), sending the recovery code to the entered email.

Composition:
- **Parent components:** App
- **Child components:** InputField, Button

Variations:

RecoverPasswordPage (Code Entry)

This page allows the user to enter the code they received in their email to reset the password.

Props:
- onRecover: {function} – function triggered when the user enters the code and submits it (required)

States:
code: {string} – stores the recovery code entered by the user (updates dynamically when typed)

Behavior:
- Clicking Recover calls onRecover(code), verifying the recovery code and allowing password reset.

Composition:
- **Parent components:** App
- **Child components:** InputField, Button

Variations:

InputField

A reusable input field component for text, password, and other form inputs.

Props:
- label: {string} – label of the input field (required)
- type: {string} – input type (text, email, password, etc.) (required, default: "text")
- value: {string} – current value of the input field (required)
- onChange: {function} – function triggered when the user types (required)

States:

Behavior:
- Allows the user to edit name, email, phone, address, city, membership type, and password.
- Calls onChange(updatedField, value) when the user types in an input field.

Composition:
- **Parent components: RegistrationPage, LoginPage, RecoverPasswordPage**
- **Child components:** None

Variations:
- Version with validation (e.g., email format)
- Version hidden (for the password).

Button

A reusable button component.

Props:
- label: {string} – button text (required)
- onClick: {function} – function triggered when the button is clicked (required)
- variant: {string} – button style (primary, secondary, danger, etc.) (optional, default: "primary")

States:

Behavior:
- Calls onClick() when clicked.

Composition:
- **Parent components: RegistrationPage, LoginPage, RecoverPasswordPage**
- **Child components: None**

Variations:
- Primary, secondary, disabled states.