Saheb Jan. 5 - Jan. 9 Report

Saheb Frontend - Features Overview

Application Features

User authentication

- Login page with email and password
- Login validation and error messages

Dashboard

- Main dashboard with welcome message showing the user's name
- Sidebar navigation with collapsible menu
- Logout in the sidebar footer

Settings

- Profile tab: edit first name, last name, username, email, preferred language (English, French, Arabic), and theme (Light/Dark)
- Security tab: change password (current, new, confirm)
- Success and error messages for actions

Internationalization

- Supports English, French, and Arabic
- Interface text changes based on selected language
- Arabic uses right-to-left layout

Theme

- Light and Dark modes
- Preference saved in user profile

Responsive design

- Works on desktop, tablet, and mobile
- Sidebar collapses to icons on smaller screens

User experience

- Forms validate inputs before submission
- Loading states during operations
- Clear success and error notifications
- Profile avatar with user initials
- Settings organized in tabs (Profile and Security)

Saheb Mobile.

Project Structure

Root Level

- `app/` - File-based routing using Expo Router. Each file represents a route (e.g., `login.tsx` -> `/login`)
  - `(drawer)/` - Nested navigation group for screens accessible via side drawer navigation
  - `_layout.tsx` - Root layout configuration
- `src/` - Main source code directory
  - `components/` - Reusable UI components
    - `ui/` - Core design system components (Button, InputField, Navbar, TaskItem, etc.)
    - `icons/` - SVG icon library used throughout the application
  - `screens/` - Feature-specific screen components organized by functionality
  - `theme/` - Design system tokens (colors, typography, spacing, radius)
  - `i18n/` - Internationalization configuration and translation files

    - `hooks/` - Custom React hooks (useTranslation)
- `assets/` - Static assets including images, logos, and icons

Screens

1. **Splash Screen** (`splash-screen/`) - Initial loading screen with app branding
2. **Language Selection** (`language-selection-screen/`) - Choose preferred language (English, French, Arabic)
3. **Onboarding** (`onboarding-screen/`) - Multi-step introduction with illustrations
4. **Sign Up** (`sign-up-screen/`) - User registration with email/password and social login
5. **Login** (`login-screen/`) - Authentication with email/password and social providers
6. **Forgot Password** (`forgot-password-screen/`) - Password recovery initiation
7. **OTP Verification** (`otp-verification-screen/`) - Security verification for password reset
8. **New Password** (`new-password-screen/`) - Password reset completion
9. **Password Reset Success** (`password-reset-success-screen/`) - Confirmation after successful reset
10. **Choose Location** (`choose-location-screen/`) - City/location selection for prayer times
11. **Activate Notifications** (`activate-notifications-screen/`) - Push notification permission request
12. **Home Screen** (`home-screen/`) - Central hub with:
    - Prayer times card with next prayer countdown
    - Daily content sections (books, videos, sermons)
    - Quick access to user program
    - KPI cards for progress tracking
13. **My Program** (`program-screen/`) - Task management with three views:
    - **Day View** - Daily task list with completion tracking
    - **Week View** - Weekly overview with horizontal calendar and progress indicators
    - **Month View** - Calendar grid with task badges and selection
    - Task items with checkboxes, prayer time associations, and completion states

Internationalization (i18n)

The application supports multiple languages with full Right-to-Left (RTL) support:

Supported Languages

- English (en-US) - Default
- French (fr-FR)
- Arabic (ar-SA) - With RTL layout support

Implementation

- **Engine**: Powered by `i18next` and `react-i18next`
- **Translation Files**: JSON files in `src/i18n/locales/` for each language
- **RTL Support**: Automatic Right-to-Left layout handling for Arabic using `I18nManager`
- **Persistence**: User's language preference is stored in AsyncStorage
- **Device Detection**: Automatically detects device language on first launch
- **Usage**: Components use the `useTranslation` hook to access localized strings

Example usage:

```
const { t } = useTranslation();
<Text>{t('home.welcome')}</Text>
```

Design System

The application uses a comprehensive token-based design system ensuring visual consistency and easy maintenance.

Theme Architecture

All design tokens are centralized in a single `theme` object exported from `@/src/theme`:

```
import { theme } from '@/src/theme';

// Colors
theme.colors.primary[400]
theme.colors.text.primary

// Typography
theme.typography.heading
theme.typography.bodyB2

// Spacing
theme.spacing.xl
theme.spacing.md

// Border Radius
theme.radius.lg
theme.radius['2xl']
```

Color System

```
export const Colors = {
  light: {
    primary: { 100: '#bfd6e5', 200: '#80adcb', 300: '#4085b2', 400: '#005c98', 500: '#004572'
},
    secondary: { 100: '#00ffff', 200: '#02d7d7', 300: '#02a9a9', 400: '#1b9bd8', 500:
'#0c74bb' },
    neutral: { 100: '#ffffff', 200: '#e8e8e8', 300: '#d2d2d2', ..., 1000: '#333333' },
    text: { primary: '#0a0a0a', secondary: '#6a7282', tertiary: '#9ca3af', inverse: '#ffffff',
placeholder: '#d1d5db' },
    background: { primary: '#ffffff', secondary: '#f9fafb' },
    border: { default: '#e5e7eb', light: '#f3f4f6', dark: '#d1d5db', focused: '#005c98' },
    // ... semantic colors (red, yellow, green)
  }
}
```

Typography

```
export const Typography = {
  heading: { fontFamily: Fonts.cairo.bold, fontSize: 24, lineHeight: 32 },
  sectionTitle: { fontFamily: Fonts.cairo.semiBold, fontSize: 18, lineHeight: 24 },
  bodyB2: { fontFamily: Fonts.cairo.black, fontSize: 16, lineHeight: 22 },
  bodyB3: { fontFamily: Fonts.cairo.regular, fontSize: 14, lineHeight: 20 },
  bodyB4: { fontFamily: Fonts.cairo.black, fontSize: 12, lineHeight: 20 },
  bodyB5: { fontFamily: Fonts.cairo.medium, fontSize: 14, lineHeight: 20 },
  button: { fontFamily: Fonts.cairo.bold, fontSize: 16, lineHeight: 24 },
  buttonSecondary: { fontFamily: Fonts.cairo.medium, fontSize: 16, lineHeight: 24 },
  caption: { fontFamily: Fonts.inter.regular, fontSize: 12, lineHeight: 16 },
  small: { fontFamily: Fonts.inter.regular, fontSize: 11, lineHeight: 16 },
  tiny: { fontFamily: Fonts.cairo.regular, fontSize: 10, lineHeight: 14 },
}
```

Spacing System

```
export const spacing = {
  none: 0,
```

```
  xs: 2,      // 2px
  sm: 4,      // 4px
  md: 8,      // 8px
  lg: 12,     // 12px
  xl: 16,     // 16px
  '2xl': 20,  // 20px
  '3xl': 24,  // 24px
  '4xl': 32,  // 32px
  '5xl': 40,  // 40px
  '6xl': 48,  // 48px
  '7xl': 56,  // 56px
} as const;
```

Border Radius

```
export const radius = {
  none: 0,
  xs: 2,      // 2px
  sm: 4,      // 4px
  md: 6,      // 6px
  lg: 8,      // 8px
  xl: 10,     // 10px
  '2xl': 12,  // 12px
  '3xl': 16,  // 16px
  full: 9999, // Circular
} as const;
```

Usage Example

```
import { theme } from '@/src/theme';

// In component styles
const styles = StyleSheet.create({
  container: {
    backgroundColor: theme.colors.background.primary,
    padding: theme.spacing.xl,
    borderRadius: theme.radius.lg,
```

```
  },
  title: {
    ...theme.typography.heading,
    color: theme.colors.text.primary,
  },
});
```

UI Components

All reusable components are located in `src/components/ui/`:

- `Button` - Primary action button with variants
- `InputField` - Form input with icon support
- `Navbar` - Top navigation bar with gradient background
- `TaskItem` - Task list item with checkbox and metadata
- `ReminderCard` - Prayer reminder card with gradient
- `PrayerCard` - Prayer times display card
- `CityButton` - Location selection button
- `CitySearchBar` - Location search input
- `SocialLoginButton` - Google/Apple login buttons
- `DrawerContent` - Side navigation drawer

Icons

40+ custom SVG icons in `src/components/icons/` including:

- Prayer time icons (Fajr, Dhuhr, Asr, Maghrib, Isha)
- Navigation icons (Menu, Back, Close)
- Social icons (Google, Apple)
- Feature icons (Home, Settings, Language, Qibla)

Saheb Backend - Features Overview

User Account Management

Users can create accounts by providing their email, password, and optional personal information like first name, last name, and username. The system ensures that each email address can only be used once. Passwords are securely stored using encryption, so even if someone gains access to the database, they cannot see the actual passwords.

Authentication and Security

The application provides secure login for both regular users and administrators. When users sign in, they receive special access tokens that allow them to use protected features. These tokens expire after a certain time for security purposes. Users can also refresh their session without having to log in again, using a special refresh token. When users want to sign out, the system properly clears their session tokens.

User Preferences

Users can customize their experience by selecting their preferred language (English, French, or Arabic) and their preferred visual theme (light mode or dark mode). These preferences are saved with their account and automatically applied when they log in.

Profile Management

Logged-in users can view their own profile information at any time. They can also update their personal details such as name, username, email, language preference, and theme preference. Additionally, users can change their password at any time by providing their current password and setting a new one.

Role-Based Access Control

The system supports two types of users: regular users and administrators. Administrators have special login endpoints and can access features that regular users cannot. The system automatically checks user roles to ensure that only authorized users can access certain features.

Security Features

The application includes several security measures to protect users and data:

- All data sent to the server is validated to ensure it meets requirements and to prevent malicious input
- The system limits how many requests can be made in a short period to prevent abuse
- Security headers are set to protect against common web vulnerabilities

- Cross-origin requests are controlled to ensure only approved websites can access the API
- All user data sent over the network is validated before being processed

API Organization

The application is organized with a clear structure where all API endpoints are prefixed with "api" and versioned (currently version 1). This allows the system to evolve and add new features while maintaining compatibility with existing applications.

Logging and Monitoring

The system keeps detailed logs of all activities, which helps identify issues and understand how the application is being used. These logs are structured in a way that makes them easy to search and analyze.

Database Integration

User information is securely stored in a database. The system automatically tracks when accounts are created and when they are last updated. Users can be found by their email address or unique identifier.

Revision #6
Created 2026-01-09 18:51:35 UTC by EL MAHDI
Updated 2026-01-09 21:34:23 UTC by EL MAHDI