ASMC Mobile App - Installation & Setup Guide

Comprehensive guide for setting up the ASMC Mobile React Native application development environment, including prerequisites, installation steps, configuration, and troubleshooting.

Table of Contents

Prerequisites

System Requirements

Minimum Requirements

- **Operating System**: macOS 10.15+ (for iOS) or Windows 10+ / Ubuntu 18.04+ (for Android)
- **RAM**: 8GB minimum, 16GB recommended
- **Storage**: 20GB free space
- **Node.js**: 18.0.0 or higher
- **npm**: 8.0.0 or higher

Recommended Requirements

- **Operating System**: macOS 12+ (for iOS) or Windows 11 / Ubuntu 20.04+ (for Android)
- **RAM**: 16GB or higher
- **Storage**: 50GB free space
- **Node.js**: 18.17.0 or higher
- **npm**: 9.0.0 or higher

Required Software

For Android Development

1. **Java Development Kit (JDK)**
   - **Version**: JDK 11 or higher
   - **Download**: Oracle JDK or OpenJDK
2. **Android Studio**
   - **Version**: Latest stable version
   - **Download**: Android Studio
   - **Components**: Android SDK, Android SDK Platform-Tools, Android SDK Build-Tools

3. **Android SDK**

    - **API Level**: 21 (Android 5.0) minimum, 34 (Android 14) target
    - **Build Tools**: 34.0.0
    - **Platform Tools**: Latest version

Universal Tools

1. **React Native CLI**

    - **Installation**: `npm install -g @react-native-community/cli`
    - **Version**: 12.0.0 or higher

2. **Git**

    - **Version**: 2.30.0 or higher
    - **Download**: [Git]

Environment Setup

Node.js Installation

Using Node Version Manager (Recommended)

```bash
# Install nvm (Node Version Manager)
curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.0/install.sh | bash

# Restart terminal or source profile
source ~/.bashrc  # or ~/.zshrc

# Install and use Node.js 18
nvm install 18.17.0
nvm use 18.17.0
nvm alias default 18.17.0

# Verify installation
node --version  # Should output v18.17.0
npm --version   # Should output 9.x.x
```

Direct Installation

```bash
# Download and install Node.js from official website
# https://nodejs.org/en/download/

# Verify installation
node --version
npm --version
```

Android Environment Setup

1. Install Android Studio

1. Download Android Studio from developer.android.com
2. Run the installer and follow the setup wizard
3. Install the following components:

    - Android SDK
    - Android SDK Platform
    - Android Virtual Device
    - Performance (Intel ® HAXM) - for Windows/macOS

2. Configure Android SDK

1. Open Android Studio
2. Go to **File > Settings** (or **Android Studio > Preferences** on macOS)
3. Navigate to **Appearance & Behavior > System Settings > Android SDK**
4. Install the following:
    - **SDK Platforms**: Android 14 (API 34), Android 13 (API 33), Android 5.0 (API 21)
    - **SDK Tools**: Android SDK Build-Tools, Android SDK Platform-Tools, Android SDK Tools

3. Set Environment Variables

Windows:

```
# Add to System Environment Variables
ANDROID_HOME=C:\Users\%USERNAME%\AppData\Local\Android\Sdk
PATH=%PATH%;%ANDROID_HOME%\platform-
tools;%ANDROID_HOME%\tools;%ANDROID_HOME%\tools\bin
```

macOS/Linux:

```
# Add to ~/.bashrc or ~/.zshrc
export ANDROID_HOME=$HOME/Android/Sdk
export PATH=$PATH:$ANDROID_HOME/emulator
export PATH=$PATH:$ANDROID_HOME/platform-tools
export PATH=$PATH:$ANDROID_HOME/tools
export PATH=$PATH:$ANDROID_HOME/tools/bin

# Apply changes
source ~/.bashrc  # or ~/.zshrc
```

4. Create Android Virtual Device (AVD)

1. Open Android Studio
2. Go to **Tools > AVD Manager**
3. Click **Create Virtual Device**
4. Select **Pixel 4** or similar device
5. Choose **API 34** (Android 14) system image
6. Configure AVD settings and click **Finish**

React Native CLI Setup

```
# Install React Native CLI globally
npm install -g @react-native-community/cli

# Verify installation
npx react-native --version

# Install additional tools
```

```
npm install -g react-devtools
npm install -g flipper
```

Project Installation

1. Clone Repository

```
# Clone the repository
git clone <repository-url>
cd asmcdae-mobile

# Check repository status
git status
git branch -a
```

2. Install Dependencies

```
# Install npm dependencies
npm install

# Verify installation
npm list --depth=0
```

3. Verify Project Structure

```
# Check project structure
ls -la

# Verify key files exist
ls -la package.json
ls -la android/
ls -la src/
```

4. Check React Native Environment

```
# Check React Native environment
npx react-native doctor

# This should show:
#  Node.js
#  Watchman (if installed)
#  yarn or npm
#  Android toolchain (if Android setup is complete)
```

Configuration

Environment Configuration

1. Create Environment Files

```
# Create environment files
touch .env
touch .env.development
touch .env.production
```

2. Configure Environment Variables

.env:

```
# API Configuration
API_BASE_URL=http://localhost:7055/api
API_TIMEOUT=30000

# App Configuration
APP_NAME=ASMC Mobile
APP_VERSION=1.0.0

# Authentication
JWT_STORAGE_KEY=asmc_mobile_token
REFRESH_TOKEN_KEY=asmc_mobile_refresh

# Feature Flags
ENABLE_DEBUG_MODE=true
ENABLE_LOGGING=true
ENABLE_ANALYTICS=false
```

.env.development:

```
# Development API
API_BASE_URL=http://10.0.2.2:7055/api

# Debug settings
ENABLE_DEBUG_MODE=true
ENABLE_LOGGING=true
ENABLE_REACT_QUERY_DEVTOOLS=true
```

.env.production:

```
# Production API
API_BASE_URL=https://api.asmc.com/api

# Production settings
ENABLE_DEBUG_MODE=false
ENABLE_LOGGING=false
ENABLE_ANALYTICS=true
```

3. Update Constants File

src/helpers/constants.js:

```
import Config from 'react-native-config';

export const baseUrl = Config.API_BASE_URL || 'http://localhost:7055/api';
export const apiTimeout = parseInt(Config.API_TIMEOUT) || 30000;
export const appName = Config.APP_NAME || 'ASMC Mobile';
export const appVersion = Config.APP_VERSION || '1.0.0';

export const isDebugMode = Config.ENABLE_DEBUG_MODE === 'true';
export const isLoggingEnabled = Config.ENABLE_LOGGING === 'true';
export const isAnalyticsEnabled = Config.ENABLE_ANALYTICS === 'true';

export const apiEndpoints = {
    auth: {
        login: '/auth/login',
        logout: '/auth/logout',
        refresh: '/auth/refresh',
        profile: '/auth/profile',
        changePassword: '/auth/change-password',
    },
    members: {
        profile: '/members/profile',
        update: '/members/update',
        family: '/members/family',
    },
    activities: {
        list: '/activities',
        detail: '/activities',
        enroll: '/activities/enroll',
        enrolled: '/activities/enrolled',
    },
    events: {
        list: '/events',
        detail: '/events',
        register: '/events/register',
        registered: '/events/registered',
    },
    halls: {
        list: '/halls',
        detail: '/halls',
        availability: '/halls/availability',
        book: '/halls/book',
        bookings: '/halls/bookings',
    },
    payments: {
        history: '/payments/history',
        process: '/payments/process',
        methods: '/payments/methods',
    },
};
```

Android Configuration

1. Update Android Manifest

android/app/src/main/AndroidManifest.xml:

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.asmc.mobile">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.CAMERA" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:name=".MainApplication"
        android:label="@string/app_name"
        android:icon="@mipmap/ic_launcher"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:allowBackup="false"
        android:theme="@style/AppTheme"
        android:usesCleartextTraffic="true">

        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"

android:configChanges="keyboard|keyboardHidden|orientation|screenSize|uiMode"
            android:launchMode="singleTask"
            android:windowSoftInputMode="adjustResize"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

2. Configure Build Settings

android/app/build.gradle:

```gradle
android {
    compileSdkVersion 34
    buildToolsVersion "34.0.0"

    defaultConfig {
        applicationId "com.asmc.mobile"
        minSdkVersion 21
        targetSdkVersion 34
        versionCode 1
        versionName "1.0.0"

        multiDexEnabled true

        // Environment-specific configuration
        buildConfigField "String", "API_BASE_URL",
"\"${project.env.get("API_BASE_URL", "http://localhost:7055/api")}\""
        buildConfigField "boolean", "DEBUG_MODE",
"${project.env.get("ENABLE_DEBUG_MODE", "true")}"
    }

    buildTypes {
        debug {
            debuggable true
            minifyEnabled false
            shrinkResources false
            buildConfigField "String", "API_BASE_URL", "\"http://10.0.2.2:7055/api\""
        }

        release {
            debuggable false
            minifyEnabled true
            shrinkResources true
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
            buildConfigField "String", "API_BASE_URL", "\"https://api.asmc.com/api\""

            signingConfig signingConfigs.release
        }
    }
}
```

3. Configure ProGuard Rules

android/app/proguard-rules.pro:

```
# React Native
-keep class com.facebook.react.** { *; }
-keep class com.facebook.jni.** { *; }

# OkHttp
-dontwarn okhttp3.**
-dontwarn okio.**
-dontwarn javax.annotation.**

# Retrofit
-dontwarn retrofit2.**
-keep class retrofit2.** { *; }
-keepattributes Signature
-keepattributes Exceptions

# Gson
-keepattributes Signature
-keepattributes *Annotation*
-dontwarn sun.misc.**
-keep class com.google.gson.** { *; }
-keep class * implements com.google.gson.TypeAdapterFactory
```

```
-keep class * implements com.google.gson.JsonSerializer
-keep class * implements com.google.gson.JsonDeserializer
```

Development Setup

1. Start Development Server

```
# Start Metro bundler
npm start

# Start with cache reset (recommended for first run)
npm run start:reset

# Start in debug mode
npm run debug
```

2. Run on Android

```
# Start Android emulator first
# Then run the app
npm run android

# Run on specific device
npx react-native run-android --deviceId=<device-id>

# Run in debug mode
npm run android:dev
```

4. Development Tools

React Native Debugger

```
# Install React Native Debugger
npm install -g react-native-debugger

# Start debugger
npm run devtools
```

Flipper Integration

1. Download Flipper from fbflipper.com
2. Install and launch Flipper
3. Enable Flipper in the app for debugging

React DevTools

```
# Install React DevTools
npm install -g react-devtools

# Start React DevTools
react-devtools
```

5. Code Quality Tools

ESLint Configuration

.eslintrc.js:

```js
module.exports = {
    root: true,
    extends: ['@react-native-community', 'plugin:react-hooks/recommended'],
    parser: '@babel/eslint-parser',
    parserOptions: {
        requireConfigFile: false,
        babelOptions: {
            presets: ['@react-native/babel-preset'],
        },
    },
    rules: {
        'react-native/no-inline-styles': 'warn',
        'react-native/no-color-literals': 'warn',
        'no-console': 'warn',
        'no-unused-vars': 'warn',
    },
    ignorePatterns: ['node_modules/', 'android/', 'ios/'],
};
```

Prettier Configuration

.prettierrc.js:

```js
module.exports = {
    arrowParens: 'avoid',
    bracketSameLine: true,
    bracketSpacing: false,
    singleQuote: true,
    trailingComma: 'all',
    tabWidth: 4,
    semi: true,
};
```

Testing Setup

1. Install Testing Dependencies

```bash
# Install testing dependencies
npm install --save-dev @testing-library/react-native @testing-library/jest-native jest

# Install additional testing utilities
npm install --save-dev react-test-renderer
```

2. Configure Jest

jest.config.js:

```
module.exports = {
    preset: 'react-native',
    setupFilesAfterEnv: ['@testing-library/jest-native/extend-expect'],
    testPathIgnorePatterns: ['/node_modules/', '/android/', '/ios/'],
    transformIgnorePatterns: [
        'node_modules/(?!(react-native|@react-native|@react-
navigation|@tanstack/react-query)/)',
    ],
    collectCoverageFrom: [
        'src/**/*.{js,jsx}',
        '!src/**/*.test.{js,jsx}',
        '!src/**/__tests__/**',
    ],
    coverageThreshold: {
        global: {
            branches: 70,
            functions: 70,
            lines: 70,
            statements: 70,
        },
    },
};
```

3. Create Test Utilities

src/tests**/test-utils.js**:

```
import React from 'react';
import { render } from '@testing-library/react-native';
import { QueryClient, QueryClientProvider } from '@tanstack/react-query';
import { NavigationContainer } from '@react-navigation/native';

const createTestQueryClient = () =>
    new QueryClient({
        defaultOptions: {
            queries: {
                retry: false,
            },
            mutations: {
                retry: false,
            },
        },
    });

export const renderWithProviders = (
    ui,
    { preloadedState = {}, queryClient = createTestQueryClient(), ...renderOptions } =
{},
) => {
    const Wrapper = ({ children }) => (
        <QueryClientProvider client={queryClient}>
            <NavigationContainer>{children}</NavigationContainer>
        </QueryClientProvider>
    );

    return render(ui, { wrapper: Wrapper, ...renderOptions });
};

export * from '@testing-library/react-native';
```

4. Write Tests

src/components/tests**/Button.test.js**:

```
import React from 'react';
import { renderWithProviders, fireEvent } from '../../__tests__/test-utils';
import Button from '../common/Button';

describe('Button Component', () => {
    it('renders correctly with title', () => {
        const { getByText } = renderWithProviders(<Button title="Test Button" />);
        expect(getByText('Test Button')).toBeTruthy();
    });

    it('calls onPress when pressed', () => {
        const onPress = jest.fn();
        const { getByText } = renderWithProviders(
            <Button title="Test Button" onPress={onPress} />,
        );

        fireEvent.press(getByText('Test Button'));
        expect(onPress).toHaveBeenCalledTimes(1);
    });

    it('shows loading state when loading prop is true', () => {
        const { getByTestId } = renderWithProviders(
            <Button title="Test Button" loading={true} />,
        );

        expect(getByTestId('loading-indicator')).toBeTruthy();
    });
});
```

5. Run Tests

```
# Run all tests
npm test

# Run tests in watch mode
npm run test:watch

# Run tests with coverage
npm run test:coverage
```

```
# Run specific test file
npm test Button.test.js
```

Production Build

Android Production Build

1. Generate Signing Key

```
# Create keystore directory
mkdir android/app/keystore

# Generate signing key
keytool -genkeypair -v -storetype PKCS12 -keystore
android/app/keystore/release.keystore -alias asmc-mobile -keyalg RSA -keysize 2048 -
validity 10000

# Enter keystore password and details when prompted
```

2. Configure Signing

android/app/build.gradle:

```
android {
    signingConfigs {
        release {
            storeFile file('keystore/release.keystore')
            storePassword 'your_store_password'
            keyAlias 'asmc-mobile'
            keyPassword 'your_key_password'
        }
    }

    buildTypes {
        release {
            signingConfig signingConfigs.release
            minifyEnabled true
            shrinkResources true
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'),
'proguard-rules.pro'
        }
    }
}
```

3. Build Release APK

```
# Clean previous builds
cd android
./gradlew clean

# Build release APK
./gradlew assembleRelease
```

```
# APK will be generated at:
# android/app/build/outputs/apk/release/app-release.apk
```

4. Build Release AAB (for Play Store)

```
# Build release AAB
./gradlew bundleRelease

# AAB will be generated at:
# android/app/build/outputs/bundle/release/app-release.aab
```

Troubleshooting

Common Issues

1. Metro Bundler Issues

Problem: Metro bundler not starting or cache issues

Solutions:

```
# Clear Metro cache
npm run start:reset

# Clear npm cache
npm cache clean --force

# Delete node_modules and reinstall
rm -rf node_modules package-lock.json
npm install

# Reset Metro cache manually
npx react-native start --reset-cache

# Kill Metro processes
npx react-native start --reset-cache --port 8081
```

2. Android Build Issues

Problem: Gradle build failures

Solutions:

```
# Clean Gradle cache
cd android
./gradlew clean

# Clear Gradle wrapper cache
rm -rf ~/.gradle/caches

# Update Gradle wrapper
./gradlew wrapper --gradle-version=8.0
```

```bash
# Check Java version
java -version  # Should be JDK 11 or higher

# Check Android SDK
echo $ANDROID_HOME
ls $ANDROID_HOME/platforms
```

4. API Connection Issues

Problem: Cannot connect to backend API

Solutions:

```bash
# Check API base URL in constants
cat src/helpers/constants.js | grep baseUrl

# Test API connection
curl http://localhost:7055/api/health

# Check network connectivity
ping localhost

# For Android emulator, use 10.0.2.2 instead of localhost
# Update API_BASE_URL to http://10.0.2.2:7055/api
```

5. Navigation Issues

Problem: Navigation not working properly

Solutions:

```bash
# Check React Navigation version
npm list @react-navigation/native

# Verify navigation structure
grep -r "createNativeStackNavigator" src/

# Check for missing dependencies
npm install @react-navigation/native @react-navigation/native-stack

# Clear navigation state
# Add this to your app initialization
import { NavigationContainer } from '@react-navigation/native';

// Reset navigation state if needed
const resetNavigationState = () => {
    // Clear navigation state logic
};
```

6. Permission Issues

Problem: App permissions not working

Solutions:

```
# Check Android permissions in manifest
cat android/app/src/main/AndroidManifest.xml | grep permission

# Grant permissions manually on device/simulator
# Android: Settings > Apps > ASMC Mobile > Permissions
```

Debug Commands

Environment Check

```
# Check React Native environment
npx react-native doctor

# Check Node.js version
node --version

# Check npm version
npm --version

# Check React Native CLI version
npx react-native --version

# Check Android SDK
echo $ANDROID_HOME
ls $ANDROID_HOME/platforms
```

Logging and Debugging

```
# Enable verbose logging
npx react-native start --verbose

# Check Metro logs
npx react-native log-android  # For Android

# Enable Chrome debugging
# Shake device or press Cmd+M (Android)
# Select "Debug" from the menu

# Use React Native Debugger
npm run devtools
```

Performance Issues

Memory Issues

```
# Check memory usage
npx react-native start --reset-cache

# Monitor app performance
# Use Flipper or React Native Debugger
# Check for memory leaks
# Use React DevTools Profiler
```

Build Performance

```
# Enable parallel builds (Android)
# Add to android/gradle.properties:
org.gradle.parallel=true
org.gradle.daemon=true
org.gradle.configureondemand=true

# Enable build cache (Android)
# Add to android/gradle.properties:
android.enableBuildCache=true
```

Summary

This comprehensive installation and setup guide covers:

- **Complete Environment Setup**: Node.js, Android Studio, and React Native CLI
- **Project Configuration**: Environment variables, build settings, and platform-specific configurations
- **Development Workflow**: Running the app, debugging, and testing
- **Production Builds**: Creating release builds for Android
- **Troubleshooting**: Common issues and their solutions

Following this guide ensures a smooth development experience for the ASMC Mobile application.

Last Updated: January 2025
Maintainer: ASMC Development Team