**🕌 ISLAMIC FINANCE COMPLIANCE FRAMEWORK**

**Kurzora Trading Platform - Shariah-Compliant Implementation**

**📋 DOCUMENT STATUS**

**Status:** ✅ **MASTER TECHNICAL SPECIFICATION**  
**Version:** 1.0  
**Authority:** Single Source of Truth for Islamic Finance Compliance  
**Implementation Priority:** Phase 2 (Months 7-12)  
**Market Opportunity:** $240M Serviceable Addressable Market

**🎯 EXECUTIVE SUMMARY**

**Vision:** Transform Kurzora into the world's most trusted Shariah-compliant trading signals platform, serving 1.8 billion Muslims globally with religiously-appropriate investment guidance.

**Strategy:** Implement comprehensive Islamic finance compliance through automated Shariah screening, religious advisory integration, and culturally-appropriate UI/UX design.

**Market Opportunity:**

* **Total Addressable Market:** $1.2B Islamic finance technology
* **Target Users:** 50M+ Islamic finance adherents globally
* **Revenue Potential:** $12M Serviceable Obtainable Market
* **Competitive Advantage:** First AI-powered Shariah-compliant signals platform

**🕌 SHARIAH COMPLIANCE REQUIREMENTS**

**Core Islamic Finance Principles**

**Prohibited Elements (Haram):**

* **Riba (Interest):** No interest-based transactions or companies
* **Gharar (Excessive Uncertainty):** No high-risk speculation
* **Maysir (Gambling):** No pure gambling or chance-based investments
* **Haram Industries:** Alcohol, pork, gambling, conventional banking, insurance, adult entertainment

**Required Elements (Halal):**

* **Asset-Backed Transactions:** Real economic activity
* **Profit-Loss Sharing:** Risk sharing between parties
* **Ethical Business Operations:** Socially responsible activities
* **Transparency:** Clear disclosure of all terms and risks

**Financial Screening Criteria**

**Debt-to-Market Cap Ratio:** ≤ 33%

* Total debt should not exceed one-third of market capitalization
* Excludes companies heavily dependent on interest-based financing

**Interest Income Ratio:** ≤ 5%

* Interest income from deposits should not exceed 5% of total income
* Ensures primary business is not interest-based

**Haram Revenue Ratio:** ≤ 5%

* Revenue from prohibited activities should not exceed 5%
* Allows minimal incidental haram income (e.g., bank interest on deposits)

**Cash-to-Market Cap Ratio:** ≤ 33%

* Cash and interest-bearing securities should not exceed one-third of market cap
* Ensures investment is in productive assets, not cash holdings

**🗄️ DATABASE IMPLEMENTATION**

**Enhanced Stock Universe Table**

-- Enhanced stock\_universe table with Islamic compliance

ALTER TABLE stock\_universe ADD COLUMN IF NOT EXISTS

-- Shariah compliance status

is\_islamic\_compliant BOOLEAN DEFAULT false,

shariah\_status VARCHAR(20) DEFAULT 'pending' CHECK (shariah\_status IN ('compliant', 'non\_compliant', 'pending', 'review\_required')),

-- Financial ratios for screening

debt\_to\_market\_cap\_ratio DECIMAL(5,2),

interest\_income\_ratio DECIMAL(5,2),

haram\_revenue\_ratio DECIMAL(5,2),

cash\_to\_market\_cap\_ratio DECIMAL(5,2),

-- Industry classification

shariah\_industry\_category VARCHAR(100),

business\_activity\_description TEXT,

-- Compliance metadata

last\_shariah\_screening\_date TIMESTAMP WITH TIME ZONE,

shariah\_screening\_source VARCHAR(100),

shariah\_board\_approval BOOLEAN DEFAULT false,

-- Advisory information

shariah\_advisor\_notes TEXT,

compliance\_confidence\_score INTEGER CHECK (compliance\_confidence\_score >= 0 AND compliance\_confidence\_score <= 100),

-- Purification requirements

requires\_income\_purification BOOLEAN DEFAULT false,

purification\_percentage DECIMAL(5,2);

-- Shariah compliance screening history

CREATE TABLE shariah\_screening\_history (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

ticker VARCHAR(10) NOT NULL REFERENCES stock\_universe(ticker),

-- Screening details

screening\_date TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

screening\_method VARCHAR(50) NOT NULL, -- 'automated', 'manual', 'advisory\_board'

previous\_status VARCHAR(20),

new\_status VARCHAR(20) NOT NULL,

-- Financial ratios at time of screening

debt\_ratio DECIMAL(5,2),

interest\_ratio DECIMAL(5,2),

haram\_revenue\_ratio DECIMAL(5,2),

cash\_ratio DECIMAL(5,2),

-- Decision factors

compliance\_factors JSONB,

non\_compliance\_reasons TEXT[],

advisory\_notes TEXT,

-- Metadata

screened\_by VARCHAR(255), -- system, advisor name, or board

confidence\_score INTEGER,

requires\_manual\_review BOOLEAN DEFAULT false,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Shariah advisory board decisions

CREATE TABLE shariah\_advisory\_decisions (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

-- Decision details

decision\_date DATE NOT NULL,

decision\_type VARCHAR(50) NOT NULL, -- 'stock\_approval', 'methodology\_update', 'industry\_ruling'

decision\_reference VARCHAR(100) UNIQUE NOT NULL,

-- Affected stocks

affected\_tickers TEXT[],

industry\_sectors TEXT[],

-- Decision content

ruling\_summary TEXT NOT NULL,

detailed\_explanation TEXT,

supporting\_evidence TEXT,

quranic\_references TEXT[],

hadith\_references TEXT[],

-- Advisory board

lead\_scholar VARCHAR(255),

participating\_scholars TEXT[],

unanimous\_decision BOOLEAN DEFAULT false,

-- Implementation

effective\_date DATE,

review\_date DATE,

implementation\_notes TEXT,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- User Islamic preferences

CREATE TABLE user\_islamic\_preferences (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

user\_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE UNIQUE,

-- Compliance preferences

strict\_shariah\_mode BOOLEAN DEFAULT true,

accept\_marginal\_compliance BOOLEAN DEFAULT false, -- Stocks with 3-5% haram revenue

require\_board\_approval BOOLEAN DEFAULT false,

-- Purification preferences

enable\_income\_purification BOOLEAN DEFAULT true,

auto\_calculate\_purification BOOLEAN DEFAULT true,

purification\_method VARCHAR(50) DEFAULT 'percentage\_based',

-- Advisory preferences

preferred\_scholar VARCHAR(255),

follow\_advisory\_board BOOLEAN DEFAULT true,

get\_purification\_reminders BOOLEAN DEFAULT true,

-- Educational preferences

show\_compliance\_explanations BOOLEAN DEFAULT true,

islamic\_finance\_learning\_mode BOOLEAN DEFAULT true,

-- Language and cultural preferences

preferred\_language VARCHAR(10) DEFAULT 'en',

date\_format VARCHAR(20) DEFAULT 'gregorian', -- 'gregorian', 'hijri', 'both'

prayer\_time\_integration BOOLEAN DEFAULT false,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

updated\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

-- Income purification tracking

CREATE TABLE income\_purification\_log (

id UUID PRIMARY KEY DEFAULT uuid\_generate\_v4(),

user\_id UUID NOT NULL REFERENCES users(id) ON DELETE CASCADE,

-- Trade reference

trade\_id UUID REFERENCES user\_trades(id),

ticker VARCHAR(10) NOT NULL,

-- Purification calculation

total\_profit DECIMAL(12,2) NOT NULL,

haram\_percentage DECIMAL(5,2) NOT NULL,

purification\_amount DECIMAL(12,2) NOT NULL,

-- Purification details

calculation\_method VARCHAR(50),

calculation\_date TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

purification\_status VARCHAR(20) DEFAULT 'pending' CHECK (purification\_status IN ('pending', 'calculated', 'donated', 'deferred')),

-- Charity information

suggested\_charities JSONB,

donation\_reference VARCHAR(255),

donation\_date DATE,

-- Notes

purification\_notes TEXT,

scholarly\_reference TEXT,

created\_at TIMESTAMP WITH TIME ZONE DEFAULT NOW()

);

**🔍 AUTOMATED SHARIAH SCREENING SYSTEM**

**Core Screening Algorithm**

// lib/shariah/complianceScreener.ts

interface ShariahScreeningResult {

ticker: string;

isCompliant: boolean;

complianceScore: number; // 0-100

screeningDetails: {

debtRatio: number;

interestRatio: number;

haramRevenueRatio: number;

cashRatio: number;

};

nonComplianceReasons: string[];

confidenceLevel: 'high' | 'medium' | 'low';

requiresManualReview: boolean;

purificationRequired: boolean;

purificationPercentage?: number;

}

interface CompanyFinancials {

totalDebt: number;

marketCap: number;

totalRevenue: number;

interestIncome: number;

haramRevenue: number;

cashAndEquivalents: number;

businessDescription: string;

primaryIndustry: string;

secondaryIndustries: string[];

}

class ShariahComplianceScreener {

private static readonly SCREENING\_THRESHOLDS = {

DEBT\_TO\_MARKET\_CAP: 0.33, // 33%

INTEREST\_INCOME: 0.05, // 5%

HARAM\_REVENUE: 0.05, // 5%

CASH\_TO\_MARKET\_CAP: 0.33, // 33%

MIN\_CONFIDENCE\_SCORE: 70 // Minimum for auto-approval

};

private static readonly PROHIBITED\_INDUSTRIES = [

'alcoholic beverages',

'tobacco',

'gambling',

'adult entertainment',

'pork products',

'conventional banking',

'conventional insurance',

'weapons manufacturing',

'interest-based lending'

];

async screenStock(ticker: string, financials: CompanyFinancials): Promise<ShariahScreeningResult> {

const result: ShariahScreeningResult = {

ticker,

isCompliant: false,

complianceScore: 0,

screeningDetails: {

debtRatio: 0,

interestRatio: 0,

haramRevenueRatio: 0,

cashRatio: 0

},

nonComplianceReasons: [],

confidenceLevel: 'high',

requiresManualReview: false,

purificationRequired: false

};

// 1. Industry screening (categorical exclusion)

const industryCheck = this.screenIndustry(financials);

if (!industryCheck.isCompliant) {

result.nonComplianceReasons.push(...industryCheck.reasons);

result.complianceScore = 0;

return result;

}

// 2. Financial ratio screening

const ratioCheck = this.screenFinancialRatios(financials);

result.screeningDetails = ratioCheck.ratios;

result.nonComplianceReasons.push(...ratioCheck.violations);

// 3. Calculate compliance score

result.complianceScore = this.calculateComplianceScore(ratioCheck, industryCheck);

// 4. Determine compliance status

result.isCompliant = result.complianceScore >= 70 && result.nonComplianceReasons.length === 0;

// 5. Check if purification is required

const purificationCheck = this.checkPurificationRequirement(financials);

result.purificationRequired = purificationCheck.required;

result.purificationPercentage = purificationCheck.percentage;

// 6. Determine confidence level and review requirements

result.confidenceLevel = this.assessConfidenceLevel(financials, ratioCheck);

result.requiresManualReview = this.requiresManualReview(result, financials);

// 7. Store screening result

await this.storeScreeningResult(result, financials);

return result;

}

private screenIndustry(financials: CompanyFinancials): { isCompliant: boolean; reasons: string[] } {

const reasons: string[] = [];

// Check primary industry

const primaryIndustryLower = financials.primaryIndustry.toLowerCase();

for (const prohibited of ShariahComplianceScreener.PROHIBITED\_INDUSTRIES) {

if (primaryIndustryLower.includes(prohibited)) {

reasons.push(`Primary industry involves prohibited activity: ${prohibited}`);

}

}

// Check secondary industries

for (const secondary of financials.secondaryIndustries) {

const secondaryLower = secondary.toLowerCase();

for (const prohibited of ShariahComplianceScreener.PROHIBITED\_INDUSTRIES) {

if (secondaryLower.includes(prohibited)) {

reasons.push(`Secondary business involves prohibited activity: ${prohibited}`);

}

}

}

// Check business description for keywords

const descriptionKeywords = this.extractProhibitedKeywords(financials.businessDescription);

if (descriptionKeywords.length > 0) {

reasons.push(`Business description contains prohibited activities: ${descriptionKeywords.join(', ')}`);

}

return {

isCompliant: reasons.length === 0,

reasons

};

}

private screenFinancialRatios(financials: CompanyFinancials): {

ratios: ShariahScreeningResult['screeningDetails'];

violations: string[];

} {

const violations: string[] = [];

// Calculate ratios

const debtRatio = financials.totalDebt / financials.marketCap;

const interestRatio = financials.interestIncome / financials.totalRevenue;

const haramRevenueRatio = financials.haramRevenue / financials.totalRevenue;

const cashRatio = financials.cashAndEquivalents / financials.marketCap;

const ratios = {

debtRatio: Math.round(debtRatio \* 10000) / 100, // Convert to percentage with 2 decimals

interestRatio: Math.round(interestRatio \* 10000) / 100,

haramRevenueRatio: Math.round(haramRevenueRatio \* 10000) / 100,

cashRatio: Math.round(cashRatio \* 10000) / 100

};

// Check violations

if (debtRatio > ShariahComplianceScreener.SCREENING\_THRESHOLDS.DEBT\_TO\_MARKET\_CAP) {

violations.push(`Debt-to-market-cap ratio (${ratios.debtRatio}%) exceeds 33% threshold`);

}

if (interestRatio > ShariahComplianceScreener.SCREENING\_THRESHOLDS.INTEREST\_INCOME) {

violations.push(`Interest income ratio (${ratios.interestRatio}%) exceeds 5% threshold`);

}

if (haramRevenueRatio > ShariahComplianceScreener.SCREENING\_THRESHOLDS.HARAM\_REVENUE) {

violations.push(`Haram revenue ratio (${ratios.haramRevenueRatio}%) exceeds 5% threshold`);

}

if (cashRatio > ShariahComplianceScreener.SCREENING\_THRESHOLDS.CASH\_TO\_MARKET\_CAP) {

violations.push(`Cash-to-market-cap ratio (${ratios.cashRatio}%) exceeds 33% threshold`);

}

return { ratios, violations };

}

private calculateComplianceScore(

ratioCheck: { ratios: any; violations: string[] },

industryCheck: { isCompliant: boolean; reasons: string[] }

): number {

// Start with perfect score

let score = 100;

// Industry violations are categorical (complete disqualification)

if (!industryCheck.isCompliant) {

return 0;

}

// Deduct points for ratio violations

const { ratios } = ratioCheck;

// Debt ratio scoring (0-25 points)

if (ratios.debtRatio > 33) {

score -= 25; // Complete violation

} else if (ratios.debtRatio > 25) {

score -= 15; // Close to threshold

} else if (ratios.debtRatio > 20) {

score -= 5; // Minor concern

}

// Interest income scoring (0-25 points)

if (ratios.interestRatio > 5) {

score -= 25;

} else if (ratios.interestRatio > 3) {

score -= 15;

} else if (ratios.interestRatio > 1) {

score -= 5;

}

// Haram revenue scoring (0-25 points)

if (ratios.haramRevenueRatio > 5) {

score -= 25;

} else if (ratios.haramRevenueRatio > 3) {

score -= 15;

} else if (ratios.haramRevenueRatio > 1) {

score -= 5;

}

// Cash ratio scoring (0-25 points)

if (ratios.cashRatio > 33) {

score -= 25;

} else if (ratios.cashRatio > 25) {

score -= 15;

} else if (ratios.cashRatio > 20) {

score -= 5;

}

return Math.max(0, score);

}

private checkPurificationRequirement(financials: CompanyFinancials): {

required: boolean;

percentage: number;

} {

const interestRatio = financials.interestIncome / financials.totalRevenue;

const haramRatio = financials.haramRevenue / financials.totalRevenue;

const totalHaramPercentage = (interestRatio + haramRatio) \* 100;

return {

required: totalHaramPercentage > 0.1, // Require purification if >0.1%

percentage: Math.round(totalHaramPercentage \* 100) / 100

};

}

private async storeScreeningResult(

result: ShariahScreeningResult,

financials: CompanyFinancials

): Promise<void> {

const { supabase } = await import('@/lib/supabase');

// Update stock\_universe table

await supabase

.from('stock\_universe')

.update({

is\_islamic\_compliant: result.isCompliant,

shariah\_status: result.isCompliant ? 'compliant' : 'non\_compliant',

debt\_to\_market\_cap\_ratio: result.screeningDetails.debtRatio,

interest\_income\_ratio: result.screeningDetails.interestRatio,

haram\_revenue\_ratio: result.screeningDetails.haramRevenueRatio,

cash\_to\_market\_cap\_ratio: result.screeningDetails.cashRatio,

last\_shariah\_screening\_date: new Date().toISOString(),

shariah\_screening\_source: 'automated\_system',

compliance\_confidence\_score: result.complianceScore,

requires\_income\_purification: result.purificationRequired,

purification\_percentage: result.purificationPercentage

})

.eq('ticker', result.ticker);

// Store in screening history

await supabase

.from('shariah\_screening\_history')

.insert({

ticker: result.ticker,

screening\_method: 'automated',

new\_status: result.isCompliant ? 'compliant' : 'non\_compliant',

debt\_ratio: result.screeningDetails.debtRatio,

interest\_ratio: result.screeningDetails.interestRatio,

haram\_revenue\_ratio: result.screeningDetails.haramRevenueRatio,

cash\_ratio: result.screeningDetails.cashRatio,

compliance\_factors: {

complianceScore: result.complianceScore,

confidenceLevel: result.confidenceLevel,

screeningVersion: '1.0'

},

non\_compliance\_reasons: result.nonComplianceReasons,

screened\_by: 'automated\_system',

confidence\_score: result.complianceScore,

requires\_manual\_review: result.requiresManualReview

});

}

private extractProhibitedKeywords(description: string): string[] {

const keywords: string[] = [];

const prohibitedKeywords = [

'alcohol', 'beer', 'wine', 'liquor', 'tobacco', 'cigarette',

'casino', 'gambling', 'lottery', 'adult entertainment', 'pornography',

'pork', 'ham', 'bacon', 'sausage', 'conventional banking',

'interest rate', 'usury', 'conventional insurance'

];

const descriptionLower = description.toLowerCase();

for (const keyword of prohibitedKeywords) {

if (descriptionLower.includes(keyword)) {

keywords.push(keyword);

}

}

return keywords;

}

}

**🎨 UI/UX IMPLEMENTATION**

**Islamic Finance UI Components**

// components/islamic/ShariahBadge.tsx

import React from 'react';

import { CheckCircle, XCircle, AlertTriangle, Clock } from 'lucide-react';

interface ShariahBadgeProps {

status: 'compliant' | 'non\_compliant' | 'pending' | 'review\_required';

score?: number;

size?: 'sm' | 'md' | 'lg';

showScore?: boolean;

}

export const ShariahBadge: React.FC<ShariahBadgeProps> = ({

status,

score,

size = 'md',

showScore = false

}) => {

const sizeClasses = {

sm: 'text-xs px-2 py-1',

md: 'text-sm px-3 py-1.5',

lg: 'text-base px-4 py-2'

};

const statusConfig = {

compliant: {

icon: CheckCircle,

bgColor: 'bg-green-100 dark:bg-green-900/20',

textColor: 'text-green-800 dark:text-green-300',

borderColor: 'border-green-200 dark:border-green-700',

label: 'Halal ✓'

},

non\_compliant: {

icon: XCircle,

bgColor: 'bg-red-100 dark:bg-red-900/20',

textColor: 'text-red-800 dark:text-red-300',

borderColor: 'border-red-200 dark:border-red-700',

label: 'Non-Halal ✗'

},

pending: {

icon: Clock,

bgColor: 'bg-yellow-100 dark:bg-yellow-900/20',

textColor: 'text-yellow-800 dark:text-yellow-300',

borderColor: 'border-yellow-200 dark:border-yellow-700',

label: 'Under Review'

},

review\_required: {

icon: AlertTriangle,

bgColor: 'bg-orange-100 dark:bg-orange-900/20',

textColor: 'text-orange-800 dark:text-orange-300',

borderColor: 'border-orange-200 dark:border-orange-700',

label: 'Needs Review'

}

};

const config = statusConfig[status];

const Icon = config.icon;

return (

<div className={`

inline-flex items-center gap-1.5 rounded-full border font-medium

${sizeClasses[size]}

${config.bgColor}

${config.textColor}

${config.borderColor}

`}>

<Icon className="w-4 h-4" />

<span>{config.label}</span>

{showScore && score !== undefined && (

<span className="font-bold">({score}%)</span>

)}

</div>

);

};

// components/islamic/ComplianceDetailsModal.tsx

interface ComplianceDetailsModalProps {

ticker: string;

isOpen: boolean;

onClose: () => void;

complianceData: ShariahScreeningResult;

}

export const ComplianceDetailsModal: React.FC<ComplianceDetailsModalProps> = ({

ticker,

isOpen,

onClose,

complianceData

}) => {

if (!isOpen) return null;

return (

<div className="fixed inset-0 bg-black/50 flex items-center justify-center z-50 p-4">

<div className="bg-white dark:bg-slate-800 rounded-lg max-w-2xl w-full max-h-[90vh] overflow-y-auto">

{/\* Header \*/}

<div className="p-6 border-b border-slate-200 dark:border-slate-700">

<div className="flex items-center justify-between">

<div>

<h3 className="text-lg font-semibold text-slate-900 dark:text-white">

Shariah Compliance Details: {ticker}

</h3>

<p className="text-sm text-slate-600 dark:text-slate-400 mt-1">

Detailed Islamic finance screening results

</p>

</div>

<ShariahBadge

status={complianceData.isCompliant ? 'compliant' : 'non\_compliant'}

score={complianceData.complianceScore}

showScore

/>

</div>

</div>

{/\* Content \*/}

<div className="p-6 space-y-6">

{/\* Financial Ratios \*/}

<div>

<h4 className="font-medium text-slate-900 dark:text-white mb-3">

📊 Financial Screening Ratios

</h4>

<div className="grid grid-cols-2 gap-4">

<RatioCard

title="Debt-to-Market Cap"

value={complianceData.screeningDetails.debtRatio}

threshold={33}

unit="%"

description="Islamic finance requires debt ≤ 33% of market cap"

/>

<RatioCard

title="Interest Income"

value={complianceData.screeningDetails.interestRatio}

threshold={5}

unit="%"

description="Interest income must be ≤ 5% of total revenue"

/>

<RatioCard

title="Haram Revenue"

value={complianceData.screeningDetails.haramRevenueRatio}

threshold={5}

unit="%"

description="Non-permissible income must be ≤ 5%"

/>

<RatioCard

title="Cash-to-Market Cap"

value={complianceData.screeningDetails.cashRatio}

threshold={33}

unit="%"

description="Cash holdings should be ≤ 33% for productive investment"

/>

</div>

</div>

{/\* Compliance Issues \*/}

{complianceData.nonComplianceReasons.length > 0 && (

<div>

<h4 className="font-medium text-red-600 dark:text-red-400 mb-3">

⚠️ Compliance Issues

</h4>

<ul className="space-y-2">

{complianceData.nonComplianceReasons.map((reason, index) => (

<li key={index} className="flex items-start gap-2 text-sm">

<XCircle className="w-4 h-4 text-red-500 mt-0.5 flex-shrink-0" />

<span className="text-slate-700 dark:text-slate-300">{reason}</span>

</li>

))}

</ul>

</div>

)}

{/\* Purification Information \*/}

{complianceData.purificationRequired && (

<div className="bg-amber-50 dark:bg-amber-900/20 p-4 rounded-lg">

<h4 className="font-medium text-amber-800 dark:text-amber-300 mb-2">

💰 Income Purification Required

</h4>

<p className="text-sm text-amber-700 dark:text-amber-400 mb-3">

This investment requires purifying {complianceData.purificationPercentage}% of profits

through charitable donation to remain Shariah-compliant.

</p>

<button className="text-sm bg-amber-600 text-white px-3 py-1.5 rounded hover:bg-amber-700 transition-colors">

Learn About Purification

</button>

</div>

)}

{/\* Educational Content \*/}

<div className="bg-blue-50 dark:bg-blue-900/20 p-4 rounded-lg">

<h4 className="font-medium text-blue-800 dark:text-blue-300 mb-2">

📚 Islamic Finance Principles

</h4>

<div className="text-sm text-blue-700 dark:text-blue-400 space-y-2">

<p><strong>Riba (Interest):</strong> Islam prohibits earning money from money itself</p>

<p><strong>Gharar (Uncertainty):</strong> Excessive speculation is discouraged</p>

<p><strong>Halal Business:</strong> Company must operate in permissible industries</p>

<p><strong>Asset-Backed:</strong> Investments should be in productive real assets</p>

</div>

</div>

</div>

{/\* Footer \*/}

<div className="p-6 border-t border-slate-200 dark:border-slate-700 flex justify-end gap-3">

<button

onClick={onClose}

className="px-4 py-2 text-slate-600 dark:text-slate-400 hover:text-slate-800 dark:hover:text-slate-200 transition-colors"

>

Close

</button>

<button className="px-4 py-2 bg-blue-600 text-white rounded hover:bg-blue-700 transition-colors">

View Full Analysis

</button>

</div>

</div>

</div>

);

};

// components/islamic/IslamicModeToggle.tsx

export const IslamicModeToggle: React.FC = () => {

const [isIslamicMode, setIsIslamicMode] = useState(false);

return (

<div className="flex items-center gap-3 p-4 bg-green-50 dark:bg-green-900/20 rounded-lg">

<div className="flex-shrink-0">

<div className="w-8 h-8 bg-green-600 rounded-lg flex items-center justify-center">

<span className="text-white text-sm">🕌</span>

</div>

</div>

<div className="flex-1">

<h4 className="font-medium text-green-800 dark:text-green-300">

Islamic Finance Mode

</h4>

<p className="text-sm text-green-600 dark:text-green-400">

Show only Shariah-compliant stocks and investments

</p>

</div>

<label className="relative inline-flex items-center cursor-pointer">

<input

type="checkbox"

checked={isIslamicMode}

onChange={(e) => setIsIslamicMode(e.target.checked)}

className="sr-only peer"

/>

<div className="w-11 h-6 bg-slate-200 peer-focus:outline-none peer-focus:ring-4 peer-focus:ring-green-300 dark:peer-focus:ring-green-800 rounded-full peer dark:bg-slate-700 peer-checked:after:translate-x-full peer-checked:after:border-white after:content-[''] after:absolute after:top-[2px] after:left-[2px] after:bg-white after:border-gray-300 after:border after:rounded-full after:h-5 after:w-5 after:transition-all dark:border-gray-600 peer-checked:bg-green-600"></div>

</label>

</div>

);

};

**🌍 ARABIC RTL SUPPORT**

**RTL Layout Implementation**

/\* styles/islamic-rtl.css \*/

/\* RTL Support for Arabic \*/

[dir="rtl"] {

direction: rtl;

text-align: right;

}

[dir="rtl"] .rtl-flip {

transform: scaleX(-1);

}

/\* Islamic-specific styling \*/

.islamic-theme {

--islamic-green: #0d7377;

--islamic-gold: #b8860b;

--islamic-blue: #1e40af;

}

/\* Arabic typography \*/

.arabic-text {

font-family: 'Noto Sans Arabic', 'Amiri', 'Arabic UI Text', sans-serif;

line-height: 1.8;

letter-spacing: 0.02em;

}

/\* Islamic calendar styling \*/

.hijri-date {

font-size: 0.875rem;

color: var(--islamic-green);

margin-bottom: 0.25rem;

}

/\* Compliance indicators \*/

.halal-indicator {

background: linear-gradient(135deg, #065f46, #10b981);

color: white;

border-radius: 0.5rem;

padding: 0.25rem 0.75rem;

font-weight: 600;

font-size: 0.75rem;

}

.haram-indicator {

background: linear-gradient(135deg, #991b1b, #ef4444);

color: white;

border-radius: 0.5rem;

padding: 0.25rem 0.75rem;

font-weight: 600;

font-size: 0.75rem;

}

/\* Prayer time integration \*/

.prayer-time-notice {

background: var(--islamic-green);

color: white;

padding: 0.75rem;

border-radius: 0.5rem;

text-align: center;

margin: 1rem 0;

}

/\* Islamic patterns (optional decorative elements) \*/

.islamic-pattern {

background-image: url("data:image/svg+xml,%3Csvg width='20' height='20' viewBox='0 0 20 20' xmlns='http://www.w3.org/2000/svg'%3E%3Cg fill='%23065f46' fill-opacity='0.05' fill-rule='evenodd'%3E%3Ccircle cx='3' cy='3' r='3'/%3E%3Ccircle cx='13' cy='13' r='3'/%3E%3C/g%3E%3C/svg%3E");

}

**Arabic Language Pack**

// locales/ar/islamic.json

{

"islamic": {

"compliance": {

"halal": "حلال",

"haram": "حرام",

"compliant": "متوافق مع الشريعة",

"non\_compliant": "غير متوافق مع الشريعة",

"under\_review": "قيد المراجعة",

"shariah\_status": "الحالة الشرعية",

"compliance\_score": "نقاط التوافق الشرعي",

"screening\_date": "تاريخ الفحص",

"advisory\_board": "مجلس الشورى الشرعية"

},

"ratios": {

"debt\_ratio": "نسبة الدين إلى القيمة السوقية",

"interest\_ratio": "نسبة دخل الفوائد",

"haram\_revenue": "نسبة الإيرادات المحرمة",

"cash\_ratio": "نسبة النقد إلى القيمة السوقية",

"threshold": "الحد المسموح",

"current\_value": "القيمة الحالية"

},

"purification": {

"required": "التطهير مطلوب",

"not\_required": "التطهير غير مطلوب",

"percentage": "نسبة التطهير",

"calculation": "حساب التطهير",

"donate\_amount": "مبلغ التبرع المطلوب",

"charity\_suggestions": "اقتراحات الجمعيات الخيرية",

"purification\_guide": "دليل التطهير"

},

"industries": {

"banking": "الخدمات المصرفية التقليدية",

"insurance": "التأمين التقليدي",

"alcohol": "المشروبات الكحولية",

"tobacco": "التبغ",

"gambling": "القمار",

"adult\_entertainment": "الترفيه للبالغين",

"pork": "منتجات الخنزير",

"weapons": "تصنيع الأسلحة"

},

"principles": {

"riba": "الربا",

"gharar": "الغرر",

"maysir": "الميسر",

"halal\_business": "التجارة الحلال",

"asset\_backed": "مدعوم بالأصول",

"profit\_sharing": "المشاركة في الأرباح والخسائر"

},

"ui": {

"islamic\_mode": "الوضع الإسلامي",

"enable\_islamic\_mode": "تفعيل الوضع الإسلامي",

"shariah\_compliant\_only": "الأسهم المتوافقة مع الشريعة فقط",

"filter\_non\_compliant": "تصفية الأسهم غير المتوافقة",

"show\_purification": "إظهار متطلبات التطهير",

"hijri\_calendar": "التقويم الهجري",

"prayer\_times": "أوقات الصلاة"

}

}

}

**🔔 ISLAMIC FINANCE ALERTS & NOTIFICATIONS**

**Shariah-Specific Alert System**

// lib/islamic/alerts.ts

interface IslamicAlert {

type: 'compliance\_change' | 'purification\_reminder' | 'haram\_stock\_alert' | 'scholarly\_update';

ticker?: string;

message: string;

severity: 'info' | 'warning' | 'critical';

islamicContext: {

arabicMessage?: string;

quranicReference?: string;

hadithReference?: string;

scholarlySource?: string;

};

}

class IslamicAlertService {

async sendComplianceChangeAlert(

userId: string,

ticker: string,

oldStatus: string,

newStatus: string,

reason: string

): Promise<void> {

const alert: IslamicAlert = {

type: 'compliance\_change',

ticker,

message: `${ticker} compliance status changed from ${oldStatus} to ${newStatus}. Reason: ${reason}`,

severity: newStatus === 'non\_compliant' ? 'critical' : 'warning',

islamicContext: {

arabicMessage: `تغيرت الحالة الشرعية للسهم ${ticker} من ${this.translateStatus(oldStatus)} إلى ${this.translateStatus(newStatus)}`,

quranicReference: newStatus === 'non\_compliant' ?

'يَا أَيُّهَا الَّذِينَ آمَنُوا اجْتَنِبُوا كَثِيرًا مِّنَ الظَّنِّ (الحجرات: 12)' : undefined

}

};

await this.dispatchAlert(userId, alert);

}

async sendPurificationReminder(

userId: string,

trades: Array<{ ticker: string; profit: number; purificationAmount: number }>

): Promise<void> {

const totalPurification = trades.reduce((sum, trade) => sum + trade.purificationAmount, 0);

const alert: IslamicAlert = {

type: 'purification\_reminder',

message: `Income purification required: $${totalPurification.toFixed(2)} from ${trades.length} trades`,

severity: 'warning',

islamicContext: {

arabicMessage: `تطهير الدخل مطلوب: ${totalPurification.toFixed(2)}$ من ${trades.length} صفقة`,

hadithReference: 'إن الله طيب لا يقبل إلا طيباً (صحيح مسلم)',

scholarlySource: 'AAOIFI Shariah Standard No. 21'

}

};

await this.dispatchAlert(userId, alert);

}

private translateStatus(status: string): string {

const translations = {

'compliant': 'متوافق مع الشريعة',

'non\_compliant': 'غير متوافق مع الشريعة',

'pending': 'قيد المراجعة',

'review\_required': 'يحتاج مراجعة'

};

return translations[status] || status;

}

private async dispatchAlert(userId: string, alert: IslamicAlert): Promise<void> {

// Send through Make.com webhook with Islamic context

const makePayload = {

type: 'islamic\_alert',

data: {

userId,

alert,

timestamp: new Date().toISOString(),

hijriDate: this.getHijriDate()

}

};

// Send to Islamic-specific Make.com scenario

await fetch(process.env.MAKE\_ISLAMIC\_WEBHOOK\_URL!, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(makePayload)

});

}

private getHijriDate(): string {

// Simple Hijri date calculation (use proper library in production)

const gregorianDate = new Date();

const hijriYear = gregorianDate.getFullYear() - 579;

return `${hijriYear} هـ`;

}

}

**📚 SHARIAH ADVISORY BOARD INTEGRATION**

**Advisory Board Management System**

// lib/islamic/advisoryBoard.ts

interface ScholarProfile {

id: string;

name: string;

nameArabic: string;

title: string;

institution: string;

qualifications: string[];

specializations: string[];

languages: string[];

photoUrl?: string;

biography: string;

biographyArabic: string;

}

interface AdvisoryDecision {

id: string;

decisionDate: Date;

type: 'stock\_approval' | 'methodology\_update' | 'industry\_ruling';

reference: string;

affectedTickers: string[];

ruling: {

summary: string;

summaryArabic: string;

detailedExplanation: string;

detailedExplanationArabic: string;

quranicReferences: string[];

hadithReferences: string[];

scholarlyConsensus: boolean;

};

scholars: {

leadScholar: string;

participants: string[];

unanimousDecision: boolean;

};

implementation: {

effectiveDate: Date;

reviewDate: Date;

notes: string;

};

}

class ShariahAdvisoryService {

private scholars: ScholarProfile[] = [

{

id: 'scholar\_1',

name: 'Dr. Muhammad Al-Tayyib',

nameArabic: 'د. محمد الطيب',

title: 'Islamic Finance Scholar',

institution: 'Al-Azhar University',

qualifications: ['PhD Islamic Jurisprudence', 'Masters Islamic Finance'],

specializations: ['Banking', 'Capital Markets', 'Islamic Economics'],

languages: ['Arabic', 'English'],

biography: 'Leading expert in Islamic finance with 20+ years experience...',

biographyArabic: 'خبير رائد في التمويل الإسلامي مع أكثر من 20 عاماً من الخبرة...'

}

// Add more scholars

];

async getAdvisoryBoardDecisions(

filters?: {

type?: string;

dateFrom?: Date;

dateTo?: Date;

affectedTicker?: string;

}

): Promise<AdvisoryDecision[]> {

const { supabase } = await import('@/lib/supabase');

let query = supabase

.from('shariah\_advisory\_decisions')

.select('\*')

.order('decision\_date', { ascending: false });

if (filters?.type) {

query = query.eq('decision\_type', filters.type);

}

if (filters?.affectedTicker) {

query = query.contains('affected\_tickers', [filters.affectedTicker]);

}

const { data, error } = await query;

if (error) throw error;

return data?.map(this.mapDecisionFromDb) || [];

}

async requestScholarReview(

ticker: string,

screeningResult: ShariahScreeningResult,

userReason?: string

): Promise<string> {

const { supabase } = await import('@/lib/supabase');

const reviewRequest = {

ticker,

request\_type: 'compliance\_review',

requested\_by: 'user\_system',

request\_reason: userReason || 'Automated screening requires manual review',

screening\_data: {

complianceScore: screeningResult.complianceScore,

ratios: screeningResult.screeningDetails,

violations: screeningResult.nonComplianceReasons

},

priority: screeningResult.complianceScore < 30 ? 'high' : 'medium',

status: 'pending',

created\_at: new Date().toISOString()

};

const { data, error } = await supabase

.from('shariah\_review\_requests')

.insert(reviewRequest)

.select('id')

.single();

if (error) throw error;

// Notify advisory board via Make.com

await this.notifyAdvisoryBoard(reviewRequest);

return data.id;

}

private async notifyAdvisoryBoard(reviewRequest: any): Promise<void> {

const makePayload = {

type: 'scholar\_review\_request',

data: {

ticker: reviewRequest.ticker,

priority: reviewRequest.priority,

reason: reviewRequest.request\_reason,

screeningData: reviewRequest.screening\_data,

requestId: reviewRequest.id

}

};

await fetch(process.env.MAKE\_SCHOLAR\_WEBHOOK\_URL!, {

method: 'POST',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(makePayload)

});

}

}

**🎯 IMPLEMENTATION ROADMAP**

**Phase 1: Foundation (Months 1-3)**

* ✅ Database schema implementation
* ✅ Basic Shariah screening algorithm
* ✅ Islamic compliance badges and UI components
* ✅ Arabic language pack and RTL support

**Phase 2: Core Features (Months 4-6)**

* ✅ Advanced screening with confidence scoring
* ✅ Income purification calculator
* ✅ Islamic mode toggle and filtering
* ✅ Shariah-specific alerts and notifications

**Phase 3: Advisory Integration (Months 7-9)**

* ✅ Shariah advisory board management
* ✅ Scholar review request system
* ✅ Decision tracking and implementation
* ✅ Educational content integration

**Phase 4: Advanced Features (Months 10-12)**

* ✅ Prayer time integration
* ✅ Hijri calendar support
* ✅ Advanced purification tracking
* ✅ Islamic finance education modules

**🔍 COMPETITIVE ADVANTAGES**

**Market Differentiation**

1. **First AI-Powered Shariah Compliance:** No existing trading platform offers automated Islamic screening with explanations
2. **Comprehensive Purification System:** Automatic calculation and tracking of required charitable donations
3. **Native Arabic Support:** Full RTL layout with culturally-appropriate design
4. **Scholar Integration:** Direct access to qualified Islamic finance scholars
5. **Educational Focus:** Teaching Islamic finance principles alongside trading

**Revenue Opportunities**

* **Premium Islamic Features:** Advanced Shariah compliance tools ($49/month)
* **Scholar Consultation Services:** One-on-one consultations ($99/session)
* **Islamic Finance Education:** Certification courses ($299)
* **White-label Solutions:** License to Islamic banks and fintech companies

**📊 SUCCESS METRICS**

**Key Performance Indicators**

* **Islamic User Adoption:** Target 25% of user base by Month 12
* **Compliance Accuracy:** >95% agreement with manual scholar reviews
* **User Satisfaction:** >4.5/5 stars for Islamic features
* **Revenue Impact:** $50K+ monthly recurring revenue from Islamic features
* **Market Penetration:** 1,000+ active Islamic finance users by Month 18

**Monitoring & Analytics**

// Track Islamic feature usage

interface IslamicAnalytics {

islamicModeActivations: number;

complianceDetailsViews: number;

purificationCalculations: number;

scholarReviewRequests: number;

arabicLanguageUsage: number;

halalStockSignals: number;

}

This comprehensive Islamic Finance Compliance Framework positions Kurzora to capture the massive underserved Islamic finance market while maintaining the highest standards of religious compliance and cultural sensitivity. The technical implementation is ready for immediate development with Cursor, providing a clear competitive advantage in the global trading signals market.