🎨 MLOps Platform - Frontend Screens & Components Specification

Complete UI/UX Implementation Guide

📱 ESSENTIAL SCREENS BREAKDOWN

1. LANDING PAGE `/`

```
Layout: Marketing layout (no dashboard)
Purpose: Convert visitors to signups - "Netflix for AI Models"

Header:
├── Logo + "MLOps Platform"
├── Navigation: Features, Pricing, Login
└── CTA Button: "Start building free"

Hero Section:
├── Headline: "Deploy AI models in minutes, not months"
├── Subline: "Complete MLOps platform for teams. From experiment to production."
├── CTA: Large "Start free trial" button
├── Demo Video: Platform walkthrough (30 seconds)
└── Social proof: "Trusted by 500+ ML teams"

Features Section (4 columns):
├── 🤖 "Model Registry" - Version control for AI models
├── 🚀 "One-click Deploy" - Production deployment in minutes
├── 📊 "Real-time Monitoring" - Performance and drift detection
└── 👥 "Team Collaboration" - Multi-tenant workspace

Use Cases Section:
├── ML Engineers: "Focus on models, not infrastructure"
├── Data Scientists: "From notebook to production seamlessly"
├── Engineering Managers: "Govern AI with confidence"
└── CTOs: "Scale AI across your organization"

Pricing Section:
├── 3 tiers: Starter (Free), Professional ($199/user/month), Enterprise (Custom)
├── Feature comparison matrix
├── Annual discount toggle (20% off)
└── FAQ accordion

Footer:
├── Links: Privacy, Terms, Support, Blog, API Docs
├── Social media: GitHub, LinkedIn, Twitter
└── Company info and contact
```

2. AUTHENTICATION PAGES

2.1 Sign Up Page `/auth/signup`

```
Layout: Split-screen (left: form, right: value proposition)
Purpose: User registration with organization setup

Left Panel - Registration Form:
├── Headline: "Start your MLOps journey"
├── Form Fields:
│   ├── Full Name (text input)
│   ├── Work Email (email input with validation)
│   ├── Password (password input with strength meter)
│   ├── Organization Name (text input)
│   └── Role: Dropdown (ML Engineer, Data Scientist, Manager, Other)
├── Terms checkbox: "I agree to Terms of Service and Privacy Policy"
├── Submit button: "Create account"
└── Login link: "Already have an account? Sign in"

Right Panel - Value Proposition:
├── "Join 500+ ML teams using our platform"
├── Feature highlights with icons
├── Customer testimonials carousel
└── Trust badges (SOC2, GDPR compliant)

OAuth Options:
├── Google SSO: "Continue with Google"
├── Microsoft SSO: "Continue with Microsoft"
└── GitHub SSO: "Continue with GitHub"

Features:
├── Real-time email validation
├── Password strength indicator
├── Organization slug auto-generation
├── Loading states and error handling
└── Email verification flow
```

2.2 Sign In Page `/auth/signin`

```
Layout: Centered form with minimal design
Purpose: User authentication

Form Fields:
├── Email (email input with validation)
├── Password (password input)
├── Remember me checkbox
└── Submit button: "Sign in to your workspace"

Alternative Options:
├── OAuth buttons: Google, Microsoft, GitHub
├── Divider: "or continue with email"
├── Forgot password link
└── Signup link: "New to MLOps Platform? Create account"

Features:
├── Form validation with error states
├── Loading spinner during authentication
├── Auto-redirect to dashboard after login
├── Organization selection (if user belongs to multiple)
└── "Welcome back" personalization
```

3. DASHBOARD LAYOUT `/dashboard/*`

```
Layout: Sidebar + header + main content
Purpose: Main application workspace

Sidebar Navigation (collapsed/expanded states):
├── Organization logo + name
├── Navigation items:
│   ├── 📊 Overview (dashboard home)
│   ├── 📁 Projects
│   ├── 🤖 Models
│   ├── 🧪 Experiments
│   ├── 🚀 Deployments
│   ├── 📈 Monitoring
│   ├── 🚨 Alerts
│   ├── 👥 Team (Professional+)
│   ├── ⚙️ Settings
│   └── 💳 Billing
├── Plan indicator: "Professional Plan" with usage bar
├── Quick actions: "New Project", "Upload Model"
└── User menu: Profile, Switch Organization, Logout

Top Header:
├── Breadcrumb navigation
├── Global search bar (models, experiments, deployments)
├── Notifications bell with badge
├── Organization switcher (if multiple orgs)
├── Primary CTA: "Deploy Model" or contextual action
└── User avatar with role badge

Main Content Area:
├── Page-specific content with consistent spacing
├── Grid layout system (12-column)
├── Loading skeletons for async content
├── Empty states with helpful actions
└── Responsive breakpoints (desktop, tablet, mobile)
```

4. DASHBOARD OVERVIEW `/dashboard`

```
Purpose: Executive summary and quick access to key features

Welcome Section:
├── Personalized greeting: "Welcome back, [Name]"
├── Organization overview: Members, models, active deployments
├── Quick setup checklist for new users
└── Recent activity summary

Metrics Cards Row (4 cards with trend indicators):
├── 📊 Total Models: "24 models" (+3 this week)
├── 🚀 Active Deployments: "12 running" (2 pending)
├── 📈 Uptime: "99.2% average" (↑ 0.1% vs last week)
└── 🔔 Active Alerts: "3 warnings" (1 critical)

Recent Activity Feed (left column):
├── Model deployments, experiment completions, alerts
├── Activity timeline with user avatars
├── Real-time updates with WebSocket
├── Filter by: All, Models, Deployments, Alerts, Team
└── "View all activity" link to full audit log

Quick Actions Grid (right column):
├── "Deploy New Model" - Primary CTA
├── "Create Experiment" - Secondary action
├── "View Model Performance" - Analytics
├── "Invite Team Member" - Collaboration
├── "Set Up Monitoring" - Configuration
└── "Browse Model Registry" - Discovery

Projects Overview:
├── Recent projects grid (max 6)
├── Project cards: Name, model count, last activity
├── Progress indicators for active experiments
├── Quick access buttons: View, Deploy, Monitor
└── "View all projects" link

Performance Summary Chart:
├── Model performance trends (last 30 days)
├── Deployment success rates
├── Response time averages
├── Interactive tooltips with details
└── Export chart data option
```

5. PROJECTS MANAGEMENT

5.1 Projects List `/dashboard/projects`

```
Purpose: Organize and manage ML projects

Header Section:
├── Page title: "Projects"
├── Create project button (primary CTA)
├── Search bar: "Search projects..."
├── Filter dropdown: All, Active, Archived
└── Sort options: Name, Created date, Last activity, Models count

Project Grid Layout:
├── Card-based layout (3-4 columns)
├── Project card components:
│   ├── Project name and description
│   ├── Member avatars (max 5 shown)
│   ├── Statistics: Models count, experiments, deployments
│   ├── Last activity timestamp
│   ├── Status badge: Active, Inactive, Archived
│   └── Quick actions: View, Edit, Archive, Delete
├── Empty state: "Create your first project"
└── Pagination for large lists

Project Stats Summary:
├── Total projects count
├── Active vs. archived breakdown
├── Team projects vs. personal projects
└── Average models per project
```

5.2 Create/Edit Project `/dashboard/projects/new` & `/dashboard/projects/[id]/edit`

```
Purpose: Project configuration and setup

Form Layout:
Basic Information:
├── Project Name: "Fraud Detection System" (required)
├── Description: Rich text editor for project details
├── Visibility: Radio buttons (Private, Organization, Public)
├── Tags: Multi-select with autocomplete
└── Project avatar: Image upload with cropping

Team Configuration:
├── Project owner: Current user (non-editable on create)
├── Team members: User search and invite system
├── Role assignment: Owner, Contributor, Viewer
├── Member permissions matrix
└── Bulk invite via email list

Settings:
├── Default model environment: Development, Staging, Production
├── Experiment retention policy: Dropdown (30, 90, 365 days, Forever)
├── Notification preferences: Checkboxes for different events
├── Integration settings: Git repository, Slack channel
└── Advanced settings: Resource limits, custom domains

Form Actions:
├── Save draft button (saves without creating)
├── Create project button (primary action)
├── Cancel button (with unsaved changes warning)
└── Delete project button (edit mode only, with confirmation)
```

5.3 Project Details `/dashboard/projects/[id]`

```
Purpose: Project workspace and resource management

Project Header:
├── Project name with edit inline capability
├── Description with expand/collapse
├── Member avatars with role tooltips
├── Project settings menu (edit, archive, delete)
└── Project stats: Models, experiments, deployments

Navigation Tabs:
├── Models: Model registry for this project
├── Experiments: Experiment tracking and comparison
├── Deployments: Model deployments and endpoints
├── Monitoring: Performance dashboards
├── Settings: Project configuration
└── Activity: Project audit log

Models Tab:
├── Model grid with status indicators
├── Quick filters: Framework, type, stage
├── Bulk actions: Deploy, archive, delete
├── Upload new model button
└── Model comparison tool

Activity Tab:
├── Timeline of all project activities
├── Activity types: Model uploads, deployments, experiments
├── User attribution and timestamps
├── Filter and search capabilities
└── Export activity report
```

6. MODEL REGISTRY

6.1 Models List `/dashboard/models`

```
Purpose: Central repository for all ML models

Header and Filters:
├── Page title: "Model Registry"
├── Upload model button (primary CTA)
├── Search: "Search models by name, description, or tags"
├── Filters sidebar:
│   ├── Project: Multi-select dropdown
│   ├── Framework: scikit-learn, TensorFlow, PyTorch, XGBoost
│   ├── Model Type: Classification, Regression, NLP, Computer Vision
│   ├── Stage: Development, Staging, Production, Archived
│   └── Tags: Multi-select with popular tags
└── Sort options: Name, Created, Updated, Performance

Model Grid/Table View Toggle:
Grid View:
├── Model cards (3-4 per row)
├── Model thumbnail/icon based on framework
├── Key metrics displayed prominently
├── Quick action buttons overlay on hover
└── Favorite/star functionality

Table View:
├── Columns: Name, Framework, Type, Stage, Performance, Last Updated
├── Sortable columns with indicators
├── Row selection for bulk actions
├── Expandable rows for quick preview
└── Context menu on right-click

Model Card/Row Content:
├── Model name and version
├── Framework badge and model type
├── Current stage with colored indicator
├── Key performance metrics (accuracy, F1, etc.)
├── Last updated timestamp and user
├── Quick actions: View, Deploy, Compare, Archive
└── Tags display with color coding
```

6.2 Upload/Register Model `/dashboard/models/new`

```
Purpose: Register new ML models in the platform

Step-by-Step Wizard:

Step 1 - Basic Information:
├── Model Name: "Customer Churn Predictor" (required)
├── Description: Rich text editor with markdown support
├── Project Assignment: Dropdown with user's projects
├── Tags: Multi-select with autocomplete and creation
└── Model Category: Dropdown (Classification, Regression, etc.)

Step 2 - Model Details:
├── Framework: Radio buttons (scikit-learn, TensorFlow, PyTorch, etc.)
├── Model Type: Specific type based on framework selection
├── Version: Auto-generated or manual input (e.g., "1.0.0")
├── Stage: Radio buttons (Development, Staging, Production)
└── Model File Upload: Drag & drop area with progress bar

Step 3 - Performance Metrics:
├── Metric inputs based on model type:
│   ├── Classification: Accuracy, Precision, Recall, F1-score
│   ├── Regression: MAE, MSE, RMSE, R²
│   └── Custom metrics: Key-value pairs
├── Training Details:
│   ├── Training dataset info
│   ├── Training duration
│   ├── Hardware used
│   └── Training logs upload (optional)
└── Validation results upload

Step 4 - Model Schema:
├── Input Schema: JSON schema definition or auto-detection
├── Output Schema: Expected output format
├── Example Input: Sample request for testing
├── Example Output: Expected response format
└── Schema validation and testing

Step 5 - Dependencies & Requirements:
├── Requirements.txt upload or manual entry
├── Python version specification
├── System requirements and resource needs
├── Environment variables needed
└── Additional dependencies or setup notes
```

```
Form Actions:
├── Previous/Next step navigation
├── Save as draft (available at any step)
├── Register model (final step)
├── Test model (validate upload and schema)
└── Cancel with confirmation dialog
```

6.3 Model Details `/dashboard/models/[id]`

```
Purpose: Comprehensive model information and management

Model Header:
├── Model name with inline editing
├── Current version and stage badges
├── Model status: Healthy, Warning, Error
├── Quick actions: Deploy, Download, Archive, Delete
├── Favorite/star toggle
└── Last activity timestamp

Model Information Tabs:

Overview Tab:
├── Model description and documentation
├── Performance metrics visualization
├── Training information and datasets used
├── Model lineage and relationships
├── Recent activity and deployment history
└── Comments and annotations system

Versions Tab:
├── Version history table
├── Version comparison tool
├── Performance metrics across versions
├── Deployment status per version
├── Version promotion/demotion actions
└── Version notes and changelog

Schema Tab:
├── Input/output schema visualization
├── Interactive API explorer
├── Example requests and responses
├── Schema validation testing
├── Download schema definitions
└── Integration code snippets

Performance Tab:
├── Model performance metrics over time
├── Comparison with baseline models
├── Performance across different datasets
├── A/B testing results (if applicable)
├── Performance degradation alerts
└── Metric definitions and explanations
```

```
Deployment Tab:
├── Current deployments across environments
├── Deployment configuration and settings
├── Resource usage and scaling information
├── Endpoint URLs and access information
├── Deployment logs and monitoring
└── Quick deploy to different environments
```

7. EXPERIMENT TRACKING

7.1 Experiments List `/dashboard/experiments`

```
Purpose: Track and compare ML experiments

Header Section:
├── Page title: "Experiments"
├── Create experiment button
├── Filter panel toggle
├── View options: List, Grid, Comparison
└── Bulk actions: Archive, delete, compare

Filter Panel:
├── Project: Multi-select dropdown
├── Status: Running, Completed, Failed, Cancelled
├── Framework: ML framework filter
├── Date range: Created/completed date picker
├── Tags: Multi-select tag filter
├── Author: Team member filter
└── Custom metric filters (accuracy > 0.9, etc.)

Experiment List:
├── Experiment name and description
├── Status badge with progress indicator
├── Key metrics preview (top 3 metrics)
├── Run count and duration
├── Author and creation date
├── Tags and project association
├── Quick actions: View, Compare, Archive, Clone
└── Checkbox for bulk selection

Comparison Mode:
├── Side-by-side experiment comparison
├── Metric comparison charts
├── Parameter difference highlighting
├── Performance trend analysis
├── Export comparison report
└── Select best performing experiment
```

7.2 Create Experiment `/dashboard/experiments/new`

```
Purpose: Set up new ML experiments

Experiment Configuration:

Basic Information:
├── Experiment Name: "Hyperparameter Tuning v3"
├── Description: Markdown-enabled description
├── Project: Dropdown selection
├── Associated Model: Optional model linkage
└── Tags: Multi-select with creation

Configuration:
├── Framework Selection: Auto-detect or manual selection
├── Hyperparameter Grid:
│   ├── Parameter name and type
│   ├── Value ranges or discrete values
│   ├── Distribution type (uniform, log-uniform, etc.)
│   └── Dynamic parameter addition
├── Objective Metrics: Primary and secondary metrics to optimize
├── Early Stopping: Criteria and patience settings
└── Resource Limits: Max runtime, compute resources

Data Configuration:
├── Training Dataset: File upload or data source connection
├── Validation Dataset: Optional validation set
├── Test Dataset: Optional test set
├── Data preprocessing: Pipeline configuration
└── Feature selection and engineering

Execution Settings:
├── Number of runs: Single run or multiple iterations
├── Parallel execution: Number of concurrent runs
├── Resource allocation: CPU, memory, GPU requirements
├── Environment: Development, staging environment selection
└── Notification settings: Success/failure notifications

Form Actions:
├── Save as template (for reuse)
├── Start experiment (begin execution)
├── Schedule experiment (future execution)
├── Validate configuration (dry run)
└── Cancel with draft saving
```

7.3 Experiment Details /dashboard/experiments/[id]

```
Purpose: Monitor and analyze experiment results

Experiment Header:
├── Experiment name and status
├── Progress bar and runtime information
├── Key metrics summary
├── Control actions: Stop, restart, clone
├── Export results button
└── Share experiment link

Tabs Navigation:

Overview Tab:
├── Experiment summary and description
├── Current best results
├── Runtime statistics and resource usage
├── Error logs and warnings
├── Experiment configuration summary
└── Related experiments and models

Runs Tab:
├── All experiment runs table
├── Run comparison and filtering
├── Individual run details access
├── Bulk run operations
├── Run artifacts and outputs
└── Performance visualization

Metrics Tab:
├── Real-time metrics dashboard
├── Metric comparison across runs
├── Interactive metric plots
├── Metric correlation analysis
├── Custom metric visualization
└── Metric export functionality

Logs Tab:
├── Real-time training logs
├── Error and warning logs
├── System resource logs
├── Log search and filtering
├── Log level selection
└── Download logs functionality
```

```
Artifacts Tab:
├── Generated artifacts (models, plots, data)
├── Artifact versioning and comparison
├── Download and sharing capabilities
├── Artifact metadata and descriptions
├── Visual artifact preview
└── Artifact lineage tracking
```

8. MODEL DEPLOYMENT

8.1 Deployments List `/dashboard/deployments`

```
Purpose: Manage model deployments across environments

Deployment Overview:
├── Environment tabs: All, Development, Staging, Production
├── Deployment status dashboard
├── Resource usage summary
├── Quick deployment button
└── Bulk management actions

Deployment Grid:
├── Deployment cards with status indicators
├── Model name and version information
├── Environment and endpoint details
├── Health status and uptime metrics
├── Resource usage (CPU, memory, requests)
├── Last deployment and update timestamps
├── Quick actions: View, Update, Scale, Terminate
└── Alert indicators for issues

Deployment Filters:
├── Status: Active, Inactive, Deploying, Failed
├── Environment: Development, Staging, Production
├── Model: Filter by specific models
├── Health: Healthy, Warning, Critical
└── Date range: Deployment date filtering

Deployment Actions:
├── Deploy new model (primary CTA)
├── Bulk operations: Scale, restart, terminate
├── Health check all deployments
├── Export deployment report
└── Deployment templates management
```

8.2 Deploy Model `/dashboard/deployments/new`

```
Purpose: Deploy ML models to production environments

Deployment Wizard:

Step 1 - Model Selection:
├── Model browser with search and filters
├── Model version selection
├── Performance metrics display
├── Model compatibility check
├── Deployment readiness validation
└── Model preview and testing

Step 2 - Environment Configuration:
├── Environment selection: Development, Staging, Production
├── Deployment name and description
├── Resource allocation:
│   ├── Instance type: CPU, GPU options
│   ├── Memory and storage requirements
│   ├── Scaling configuration (min/max instances)
│   └── Auto-scaling policies
├── Network configuration:
│   ├── Endpoint URL customization
│   ├── Security groups and access control
│   ├── Load balancer settings
│   └── SSL certificate configuration
└── Environment variables and secrets

Step 3 - Deployment Configuration:
├── Container settings:
│   ├── Base image selection
│   ├── Dependencies and requirements
│   ├── Health check endpoints
│   └── Startup and readiness probes
├── Monitoring configuration:
│   ├── Metrics collection settings
│   ├── Logging configuration
│   ├── Alert thresholds
│   └── Performance monitoring
└── Rollback and update policies

Step 4 - Review and Deploy:
├── Configuration summary
├── Cost estimation
├── Deployment timeline estimate
├── Pre-deployment validation
├── Test deployment option
└── Scheduled deployment settings

Form Actions:
├── Previous/Next navigation
├── Save as template
├── Test configuration
├── Deploy now (primary action)
└── Schedule deployment
```

8.3 Deployment Details `/dashboard/deployments/[id]`

```
Purpose: Monitor and manage individual deployments

Deployment Header:
├── Deployment name and status
├── Model and version information
├── Environment and endpoint URL
├── Health status indicator
├── Last update timestamp
└── Quick actions: Update, Scale, Restart, Terminate

Monitoring Dashboard:

Performance Metrics:
├── Real-time request rate and latency
├── Error rate and success rate
├── Resource utilization (CPU, memory, GPU)
├── Response time distribution
├── Throughput and concurrent requests
└── Custom business metrics

Health Monitoring:
├── Service health status
├── Dependency health checks
├── Infrastructure health
├── Data quality monitoring
├── Model performance drift
└── Alert status and history

Logs and Debugging:
├── Real-time application logs
├── Error logs and stack traces
├── Request/response logs
├── System logs
├── Log search and filtering
└── Log export functionality

Configuration Tab:
├── Current deployment configuration
├── Environment variables
├── Scaling policies
├── Network and security settings
├── Update deployment configuration
└── Configuration history
```

```
History Tab:
├── Deployment history and changes
├── Rollback options
├── Performance over time
├── Incident timeline
├── Configuration changes log
└── User activity audit
```

9. MONITORING & ALERTS

9.1 Monitoring Dashboard `/dashboard/monitoring`

```
Purpose: Centralized monitoring for all models and deployments

Overview Metrics:
├── System-wide health score
├── Total active deployments
├── Alert summary (critical, warning, info)
├── Overall uptime percentage
├── Average response time across all models
└── Resource utilization summary

Model Performance Grid:
├── Performance cards for each deployed model
├── Key metrics: Accuracy, latency, throughput
├── Status indicators: Healthy, Warning, Critical
├── Trend indicators: Improving, stable, degrading
├── Quick drill-down to model details
└── Model comparison capabilities

Alerts Section:
├── Active alerts list with priority
├── Alert timeline and history
├── Alert acknowledgment and resolution
├── Alert routing and escalation status
├── Bulk alert management
└── Create new alert rule button

Data Quality Monitoring:
├── Data drift detection results
├── Feature distribution changes
├── Data quality scores
├── Anomaly detection results
├── Input validation errors
└── Data lineage tracking

Performance Analytics:
├── System performance trends
├── Model accuracy over time
├── Response time analysis
├── Error rate tracking
├── Resource usage optimization
└── Cost analysis and optimization
```

9.2 Alert Configuration `/dashboard/alerts`

```
Purpose: Configure monitoring alerts and notification rules

Alert Rules Management:
├── Active alert rules list
├── Rule status: Enabled, disabled, triggered
├── Alert frequency and noise analysis
├── Rule performance and accuracy
├── Bulk rule management
└── Alert rule templates

Create Alert Rule:
├── Alert name and description
├── Trigger conditions:
│   ├── Metric thresholds (accuracy < 0.9)
│   ├── System conditions (high latency, errors)
│   ├── Data quality issues
│   └── Custom condition builder
├── Notification channels:
│   ├── Email recipients
│   ├── Slack channels
│   ├── Webhook integrations
│   └── PagerDuty escalation
├── Alert frequency and cooldown
├── Severity levels and priority
└── Alert suppression rules

Notification Channels:
├── Channel configuration and testing
├── Channel health and delivery status
├── Message templates and customization
├── Delivery preferences and scheduling
├── Channel usage analytics
└── Integration setup guides

Alert History:
├── Historical alert timeline
├── Alert resolution tracking
├── False positive analysis
├── Alert effectiveness metrics
├── Team response times
└── Alert report generation
```

10. TEAM & COLLABORATION

10.1 Team Management /dashboard/team

```
Purpose: Manage organization members and collaboration

Team Overview:
├── Organization member count and limits
├── Role distribution chart
├── Recent team activity
├── Pending invitations
├── Team performance metrics
└── Collaboration statistics

Members List:
├── Member cards with avatars and roles
├── Member status: Active, inactive, pending
├── Last activity timestamps
├── Permission levels and access
├── Project assignments
├── Quick actions: Edit, remove, message
└── Bulk member management

Invite Members:
├── Email-based invitation system
├── Role selection during invitation
├── Bulk invitation via CSV upload
├── Custom invitation messages
├── Invitation tracking and management
└── Organization joining approval workflow

Roles and Permissions:
├── Role definitions and capabilities
├── Permission matrix visualization
├── Custom role creation
├── Resource-based permissions
├── Role assignment and management
└── Permission audit and compliance

Team Activity:
├── Collaborative activity timeline
├── Project contribution tracking
├── Model sharing and handoffs
├── Team communication logs
├── Performance collaboration metrics
└── Team productivity analytics
```

11. SETTINGS & ADMINISTRATION

11.1 Organization Settings /dashboard/settings

```
Purpose: Configure organization-wide settings and preferences

Organization Profile:
├── Organization name and logo
├── Contact information and billing details
├── Organization description and industry
├── Timezone and locale settings
├── Organization URL and branding
└── Legal and compliance information

Security Settings:
├── Authentication requirements
├── Password policies
├── Two-factor authentication enforcement
├── Session management
├── API access policies
└── Security audit logs

Integration Settings:
├── Third-party integrations (Slack, GitHub, etc.)
├── API keys and webhooks
├── SSO configuration (SAML, OAuth)
├── Data export and import settings
├── Backup and recovery preferences
└── Integration health monitoring

Notification Preferences:
├── Organization-wide notification settings
├── Default alert channels
├── Notification frequency and scheduling
├── Email templates and branding
├── Notification compliance settings
└── Opt-out management

Resource Management:
├── Compute resource limits and quotas
├── Storage allocation and usage
├── API rate limiting
├── Cost management and budgets
├── Resource optimization recommendations
└── Performance monitoring settings
```

11.2 Billing & Subscription `/dashboard/billing`

```
Purpose: Manage subscription, usage, and payments

Current Plan Overview:
├── Plan name and pricing
├── Billing cycle and next payment
├── Current usage vs. limits
├── Plan feature comparison
├── Upgrade/downgrade options
└── Usage analytics and trends

Usage Dashboard:
├── Usage metrics by category
├── Historical usage trends
├── Cost breakdown and analysis
├── Resource utilization efficiency
├── Usage forecasting
└── Cost optimization recommendations

Billing History:
├── Invoice history and downloads
├── Payment method management
├── Billing address and tax information
├── Payment history and status
├── Refund and credit management
└── Billing dispute resolution

Plan Management:
├── Plan comparison and features
├── Upgrade/downgrade workflows
├── Usage limit management
├── Add-on services
├── Custom enterprise pricing
└── Plan change scheduling
```

🎨 DESIGN SYSTEM COMPONENTS

Core UI Components

```
Buttons: Primary, Secondary, Outline, Ghost, Destructive, Icon
Inputs: Text, Email, Password, Number, Textarea, File Upload
Selectors: Dropdown, Multi-select, Radio, Checkbox, Toggle
Feedback: Toast, Alert, Banner, Loading Spinner, Progress Bar
Navigation: Tabs, Breadcrumbs, Pagination, Sidebar, Menu
Data Display: Table, Cards, Lists, Charts, Metrics, Badges
Overlays: Modal, Popover, Tooltip, Drawer, Confirmation Dialog
Status: Health Indicator, Progress Ring, Status Badge
ML-Specific: Model Card, Metric Display, Performance Chart
```

Color System

```
Primary Brand: Blue (#2563EB) - Actions, links, highlights
Success: Green (#059669) - Success states, healthy status
Warning: Amber (#D97706) - Warnings, attention needed
Error: Red (#DC2626) - Errors, critical alerts, dangerous actions
Info: Sky Blue (#0284C7) - Information, neutral states
Neutral: Gray scale (Slate 50-900) - Text, borders, backgrounds

Semantic Colors:
Model Healthy: Green (#10B981)
Model Warning: Yellow (#F59E0B)
Model Critical: Red (#EF4444)
Development: Blue (#3B82F6)
Staging: Orange (#F97316)
Production: Green (#10B981)
```

Typography Scale

```
Display Large: 48px (3rem) - Landing page headlines
Display: 36px (2.25rem) - Major page titles
Heading 1: 30px (1.875rem) - Section headers
Heading 2: 24px (1.5rem) - Subsection headers
Heading 3: 20px (1.25rem) - Component headers
Body Large: 18px (1.125rem) - Important body text
Body: 16px (1rem) - Default body text
Body Small: 14px (0.875rem) - Secondary text
Caption: 12px (0.75rem) - Labels, metadata
Code: Monospace - Code snippets, IDs
```

Spacing System

```
Base unit: 4px
xs: 4px - Tight spacing
sm: 8px - Small spacing
md: 16px - Medium spacing
lg: 24px - Large spacing
xl: 32px - Extra large spacing
2xl: 48px - Section spacing
3xl: 64px - Page spacing
```

Component Specifications

Model Card Component

```
Purpose: Display model information in grid/list views
Variants: Compact, detailed, comparison
States: Default, hover, selected, loading, error

Content:
├── Model name and description
├── Framework badge
├── Performance metrics
├── Status indicator
├── Action buttons
└── Metadata (author, date, tags)

Interactions:
├── Click to view details
├── Hover for quick actions
├── Select for bulk operations
├── Favorite/star toggle
└── Context menu
```

Deployment Status Component

```
Purpose: Show deployment health and status
Variants: Compact, detailed, dashboard
States: Healthy, warning, critical, deploying, inactive

Content:
├── Status indicator (color-coded)
├── Health metrics
├── Uptime information
├── Error count
└── Last check timestamp

Real-time updates:
├── WebSocket connection for live updates
├── Auto-refresh intervals
├── Status change animations
└── Alert integration
```

Metric Display Component

```
Purpose: Show model performance metrics
Variants: Single metric, metric group, comparison
States: Current, historical, target comparison

Content:
├── Metric name and value
├── Trend indicator (↑↓ with percentage)
├── Target/threshold comparison
├── Sparkline chart (optional)
├── Time period context
└── Drill-down capability

Styling:
├── Large number display
├── Color coding based on performance
├── Icon representation of metric type
├── Responsive sizing
└── Accessibility compliance
```

Chart Components

```
Performance Chart:
├── Line charts for time series data
├── Bar charts for categorical comparisons
├── Scatter plots for correlation analysis
├── Heatmaps for feature importance
└── Custom ML-specific visualizations

Chart Features:
├── Interactive tooltips
├── Zoom and pan functionality
├── Time range selection
├── Multiple metric overlay
├── Export to PNG/SVG
├── Real-time data updates
└── Responsive design

Chart Configurations:
├── Color themes (light/dark mode)
├── Animation preferences
├── Axis customization
├── Legend positioning
└── Accessibility features
```

Mobile Responsive Design

Breakpoints

```
Mobile: < 768px
Tablet: 768px - 1024px
Desktop: > 1024px
Large Desktop: > 1440px
```

Mobile Navigation

```
< 768px adaptations:
├── Hamburger menu replaces sidebar
├── Bottom navigation for key actions
├── Swipe gestures for navigation
├── Collapsible content sections
├── Touch-friendly button sizes (44px minimum)
├── Full-screen modals instead of popovers
└── Sticky headers with key actions
```

Mobile Component Adaptations

```
Dashboard:
├── Stack metric cards vertically
├── Simplified navigation
├── Condensed information display
├── Touch-optimized interactions
└── Progressive disclosure

Tables:
├── Horizontal scroll for wide tables
├── Card layout alternative
├── Sticky columns for important data
├── Expandable rows for details
└── Bulk action drawer

Forms:
├── Single column layout
├── Larger input fields
├── Step-by-step wizards
├── Simplified validation
└── Touch keyboard optimization

Charts:
├── Simplified chart types
├── Touch interactions (pinch, zoom)
├── Full-screen chart view
├── Swipe between chart types
└── Simplified legends
```

📋 IMPLEMENTATION CHECKLIST

Phase 1: Authentication & Core Layout

- □ Landing page with value proposition
- □ Sign up/sign in forms with validation
- □ Dashboard layout with sidebar navigation
- □ User profile and organization management
- □ Basic responsive design implementation

Phase 2: Project & Model Management

- □ Project CRUD operations
- □ Model registry with upload functionality
- □ Model versioning and metadata management
- □ Search and filtering capabilities
- □ Team collaboration features

Phase 3: Experiment Tracking

- □ Experiment creation and configuration
- □ Run tracking and metrics visualization
- □ Experiment comparison tools
- □ Real-time progress monitoring
- □ Artifact management system

Phase 4: Deployment & Monitoring

- □ Model deployment wizard
- □ Deployment management interface
- □ Real-time monitoring dashboards
- □ Alert configuration and management
- □ Performance analytics

Phase 5: Advanced Features

- □ Team management and permissions
- □ Billing and subscription management
- □ Advanced settings and integrations
- □ Mobile optimization
- □ Accessibility compliance

Phase 6: Polish & Launch

- □ Design system consistency

- □ Performance optimization
- □ Error handling and edge cases
- □ User onboarding flows
- □ Documentation and help system

🎨 UI/UX PRINCIPLES

Usability Guidelines

```
Clarity:
├── Clear visual hierarchy
├── Consistent terminology
├── Obvious interactive elements
├── Helpful error messages
└── Contextual guidance

Efficiency:
├── Minimal clicks to complete tasks
├── Keyboard shortcuts for power users
├── Bulk operations for repetitive tasks
├── Smart defaults and suggestions
└── Quick search and filtering

Feedback:
├── Immediate response to user actions
├── Progress indicators for long operations
├── Success/error state communication
├── Loading states for async operations
└── Real-time data updates where relevant
```

Accessibility Requirements

```
WCAG 2.1 AA Compliance:
├── Color contrast ratios > 4.5:1
├── Keyboard navigation support
├── Screen reader compatibility
├── Alt text for images and icons
├── Focus indicators
├── Semantic HTML structure
└── ARIA labels where appropriate

Additional Considerations:
├── High contrast mode support
├── Font size preferences
├── Motion reduction preferences
├── Voice control compatibility
└── Mobile accessibility
```

Performance Considerations

```
Load Time Optimization:
├── Code splitting by route
├── Lazy loading for heavy components
├── Image optimization and lazy loading
├── Bundle size monitoring
└── CDN utilization for static assets

Runtime Performance:
├── Virtual scrolling for large lists
├── Debounced search inputs
├── Optimistic UI updates
├── Efficient re-rendering patterns
└── Memory leak prevention

Data Loading:
├── Progressive loading strategies
├── Caching for frequently accessed data
├── Background data synchronization
├── Offline capability for core features
└── Error retry mechanisms
```

🔧 TECHNICAL IMPLEMENTATION NOTES

State Management

```
Global State (React Context):
├── User authentication state
├── Organization context
├── Theme preferences
├── Notification system
└── Global loading states

Component State (useState/useReducer):
├── Form data and validation
├── UI state (modals, dropdowns)
├── Local component preferences
├── Temporary user interactions
└── Component-specific loading states

Server State (React Query):
├── API data caching
├── Background synchronization
├── Optimistic updates
├── Error handling and retries
└── Pagination and infinite queries
```

API Integration Patterns

```
Data Fetching:
├── Custom hooks for API calls
├── Automatic error boundary integration
├── Loading state management
├── Cache invalidation strategies
└── Real-time updates via WebSocket

Error Handling:
├── Global error boundary
├── API error response handling
├── User-friendly error messages
├── Retry mechanisms
├── Offline state handling
└── Error reporting integration

Optimizations:
├── Request deduplication
├── Background cache updates
├── Prefetching for anticipated actions
├── Compression and minification
└── Request/response interceptors
```

Component Architecture

```
Component Structure:
├── Atomic design principles
├── Compound component patterns
├── Render props for flexibility
├── Custom hooks for logic reuse
└── Higher-order components for cross-cutting concerns

Styling Approach:
├── Tailwind CSS utility classes
├── Component variants via class variance authority
├── Custom CSS for complex layouts
├── CSS-in-JS for dynamic styles
└── Design tokens for consistency

Testing Strategy:
├── Unit tests for utility functions
├── Component testing with React Testing Library
├── Integration tests for user flows
├── Visual regression testing
└── Accessibility testing automation
```

🚀 DEPLOYMENT & MONITORING

Frontend Deployment

```
Build Process:
├── Next.js static generation where possible
├── Bundle optimization and tree shaking
├── Environment-specific builds
├── Source map generation for debugging
└── Asset optimization and compression

Hosting Strategy:
├── CDN deployment for static assets
├── Edge deployment for global performance
├── Progressive Web App capabilities
├── Service worker for offline functionality
└── Analytics and performance monitoring integration

Quality Assurance:
├── Automated testing in CI/CD pipeline
├── Cross-browser compatibility testing
├── Performance benchmarking
├── Security scanning
└── Accessibility auditing
```

Monitoring & Analytics

```
User Experience Monitoring:
├── Core Web Vitals tracking
├── User journey analysis
├── Error tracking and reporting
├── Performance monitoring
└── A/B testing infrastructure

Business Metrics:
├── Feature usage analytics
├── Conversion funnel tracking
├── User engagement metrics
├── Retention analysis
└── Custom event tracking

Technical Monitoring:
├── Application performance monitoring
├── Error rate and exception tracking
├── API response time monitoring
├── Resource usage tracking
└── Security incident detection
```

This comprehensive frontend specification provides everything needed to implement a world-class MLOps platform interface that serves ML Engineers, Data Scientists, and Engineering Managers with an intuitive, powerful, and scalable user experience! 🎨 ✨