Plan

Sheba Retention AI - Complete System Plan

Architecture Overview

Tech Stack:

- Backend: FastAPI (Python) with ML models
- Frontend: React.js with modern UI components
- Database: PostgreSQL with Redis caching
- ML: XGBoost, Scikit-learn, SHAP
- Deployment: Heroku/Railway (free tier)

System Components

1. Data Layer

- **Synthetic Data Generator** (`backend/data/synthetic_generator.py`)
- Generate 1M+ booking records (3 years history)
- Customer profiles with demographics, behavior patterns
- Service categories (AC, electrician, plumber, etc.)
- Seasonal patterns (Eid, summer AC demand)
- Competitor search events, complaints, ratings
- **Mock API Endpoints** (`backend/api/mock_sheba_api.py`)
- `/api/bookings` - Historical booking data
- `/api/customers` - Customer profiles
- `/api/services` - Service catalog
- `/api/providers` - Service provider data

2. ML Engine

Churn Prediction Model (`backend/ml/churn_model.py`)

- Features: days_since_last_booking, num_bookings, avg_spent, complaints_count, searched_competitor, service_type, loyalty_tier, seasonal_activity
- Algorithm: XGBoostClassifier
- Output: Churn probability (0-1), risk category (Low/Medium/High/Critical), SHAP explanations
- Target: 80-85% accuracy
- Save trained model as `.pkl` for FastAPI loading

Customer Segmentation (`backend/ml/segmentation_model.py`)

- K-Means clustering on: booking_value, frequency, service_diversity, price_sensitivity, quality_focus
- Segments: High-Value Loyal, Price-Sensitive, Quality-Focused, Occasional, At-Risk
- Dynamic re-segmentation monthly

Lifetime Value Predictor (`backend/ml/ltv_model.py`)

- XGBoost regression predicting 3-year customer value
- Features: historical_spend, booking_frequency, service_types, tenure, engagement_score
- Output: Predicted LTV in Tk, confidence interval

Loyalty Tier System (`backend/ml/loyalty_engine.py`)

- AI-based tier assignment: Bronze/Silver/Gold/Platinum
- Criteria: LTV, booking frequency, tenure, engagement
- Automatic tier upgrades/downgrades with benefit adjustments

Intervention Recommendation Engine (`backend/ml/intervention_engine.py`)

- Collaborative filtering for personalized offers

- Rules engine combining churn risk + segment + LTV
- Output: Recommended action, discount amount, message template, expected retention rate, ROI

3. Core Backend Services

Prediction Service (`backend/services/prediction_service.py`)

- Daily batch prediction for all active customers
- Real-time prediction API for on-demand checks
- Store predictions in PostgreSQL with timestamps

Intervention Service (`backend/services/intervention_service.py`)

- Queue system for pending interventions
- Approval workflow (AI suggests → human approves → system executes)
- Track intervention history and outcomes

Analytics Service (`backend/services/analytics_service.py`)

- Retention rate calculations
- ROI tracking per intervention
- Cohort analysis
- Revenue recovery metrics

Notification Service (`backend/services/notification_service.py`)

- Mock SMS/Email/Push notification sender
- Template engine for personalized messages
- Bengali language support
- Delivery status tracking

Payment Service (`backend/services/payment_service.py`)

- Mock bKash/Nagad/Bank APIs
- Discount code generation and validation
- Transaction logging

4. API Layer (`backend/api/routes/`)

Prediction Endpoints:

- `POST /api/predict/churn` - Get churn prediction for customer
- `GET /api/predictions/at-risk` - List high-risk customers
- `GET /api/predictions/dashboard` - Summary statistics

Intervention Endpoints:

- `GET /api/interventions/pending` - Pending approval queue
- `POST /api/interventions/approve` - Approve intervention
- `POST /api/interventions/reject` - Reject intervention
- `GET /api/interventions/history` - Past interventions

Customer Endpoints:

- `GET /api/customers/:id` - Customer profile with predictions
- `GET /api/customers/:id/loyalty` - Loyalty tier and benefits
- `GET /api/customers/:id/offers` - Personalized offers
- `POST /api/customers/:id/redeem` - Redeem discount code

Analytics Endpoints:

- `GET /api/analytics/retention` - Retention metrics
- `GET /api/analytics/roi` - ROI dashboard data
- `GET /api/analytics/segments` - Segment distribution
- `GET /api/analytics/revenue` - Revenue recovery tracking

Admin Endpoints:

- `POST /api/admin/retrain` - Trigger model retraining
- `GET /api/admin/model-performance` - Model metrics
- `POST /api/admin/settings` - Update system settings

5. Frontend Application

Admin Dashboard (`frontend/src/pages/admin/`)

- Real-time churn alerts with customer details
- Intervention approval queue with AI recommendations
- Model performance monitoring (accuracy, precision, recall)
- System settings and configuration
- Bulk intervention campaigns

Analyst Dashboard (`frontend/src/pages/analyst/`)

- Customer segmentation visualization
- Cohort analysis and retention curves
- ROI tracking and revenue recovery
- A/B testing results for interventions
- Exportable reports (PDF/CSV)

Customer Portal (`frontend/src/pages/customer/`)

- Loyalty tier display with progress bar
- Personalized benefits and offers
- Service history and upcoming reminders
- Discount code redemption
- Preferred provider selection

Shared Components (`frontend/src/components/`)

- `ChurnRiskCard` - Visual risk indicator with SHAP explanations
- `InterventionCard` - Intervention details with ROI projection
- `LoyaltyBadge` - Tier badge with benefits tooltip
- `SegmentChart` - Customer segment distribution
- `RetentionMetrics` - KPI dashboard widgets
- `CustomerProfile` - Comprehensive customer view

6. Database Schema (`backend/db/models.py`)

Tables:

- `customers` - Customer profiles
- `bookings` - Booking history
- `predictions` - Churn predictions with timestamps
- `interventions` - Intervention queue and history
- `loyalty_tiers` - Tier definitions and benefits
- `customer_loyalty` - Customer tier assignments
- `segments` - Segment definitions
- `customer_segments` - Customer segment assignments
- `notifications` - Notification log
- `model_performance` - Model metrics over time

7. Explainability & Insights

SHAP Integration (`backend/ml/explainability.py`)

- Generate SHAP values for each prediction
- Top 5 churn drivers per customer
- Feature importance visualization
- Human-readable explanations in Bengali

8. Deployment & Configuration

Docker Setup:

- `Dockerfile` - Multi-stage build for backend
- `docker-compose.yml` - Backend + PostgreSQL + Redis
- `frontend/Dockerfile` - React production build

Environment Configuration:

- `.env.example` - Template for environment variables

- `backend/config.py` - Configuration management
- Database connection strings, API keys, model paths

Deployment Scripts:

- `deploy/heroku.sh` - Heroku deployment script
- `deploy/railway.sh` - Railway deployment script
- `requirements.txt` - Python dependencies
- `frontend/package.json` - Node.js dependencies

9. Testing & Documentation

Testing:

- `tests/test_models.py` - ML model unit tests
- `tests/test_api.py` - API endpoint tests
- `tests/test_services.py` - Service layer tests
- Synthetic data validation

Documentation:

- `docs/API.md` - API documentation
- `docs/MODELS.md` - ML model specifications
- `docs/DEPLOYMENT.md` - Deployment guide
- `docs/USER_GUIDE.md` - User manual (Bengali + English)
- `README.md` - Project overview and setup

10. Monitoring & Logging

Logging Setup (`backend/utils/logger.py`)

- Structured logging for all services
- Prediction logging for model monitoring
- Intervention outcome tracking

Monitoring Dashboard:

- Model drift detection
- API response times
- Error rate tracking
- Daily prediction counts

Key Features Implementation

Churn Prevention Flow

1. Daily batch job runs predictions on all customers
2. High-risk customers (>60% churn probability) flagged
3. Intervention engine generates personalized recommendations
4. Admin/analyst reviews and approves interventions
5. System sends notifications with offers
6. Track redemption and retention outcomes

Personalization Examples

- **High-Value Drifting:** "Rahima, we miss you! Get Tk 500 off your next 3 bookings"
- **Price-Sensitive:** "New budget service category - AC cleaning from Tk 299"
- **Quality-Focused:** "Book verified premium providers with 4.8+ ratings"
- **Seasonal Reminder:** "Your AC needs pre-summer servicing - book now"

ROI Calculation

- Cost per intervention (discount + operational cost)
- Expected retention rate from historical data
- Predicted customer LTV
- Net benefit = (LTV × retention_rate) - intervention_cost

Success Metrics

Technical:

- Churn model accuracy: 80-85%
- API response time: <200ms
- System uptime: 99%+

Business:

- Retention rate improvement: 35% → 45%
- ROI per intervention: >500%
- Customer satisfaction: Track NPS changes

Development Timeline (24-Hour MVP)

Hours 0-8: Data & ML Foundation

- Generate synthetic dataset
- Train churn, segmentation, LTV models
- Validate model performance

Hours 8-16: Backend Development

- Build FastAPI application structure
- Implement prediction and intervention services
- Create mock APIs and database schema

Hours 16-24: Frontend & Integration

- Build React dashboards (admin, analyst, customer)
- Integrate frontend with backend APIs
- Deploy to Heroku/Railway
- Final testing and demo preparation