admin_manual_simple.md

UTM Campus Assistant Chatbot - Administrator Manual

Table of Contents

System Overview

The UTM Campus Assistant Chatbot is a web-based facility management system that helps students report issues, book facilities, and get information through an AI-powered chatbot.

Key Features

- **Issue Reporting**: Students can report facility problems (electrical, hygiene, structural, equipment, security)
- **Facility Booking**: Students request facility bookings, admins approve/reject
- **AI Chatbot**: Natural language interface for facility information and assistance
- **Admin Dashboard**: Complete management interface for all system operations

User Roles

- **Students**: Report issues, book facilities, use chatbot
- **Admins**: Manage issues, approve bookings, oversee system operations

Technical Components

- **Frontend**: Flask web application with Bootstrap 5 dark theme
- **Backend**: Python Flask with PostgreSQL database

- **AI Service**: DeepSeek LLM for natural language processing
- **Authentication**: Role-based access with encrypted passwords

Getting Started

Accessing the Admin Dashboard

1. **Login**: Navigate to the application URL and click "Login"
2. **Credentials**: Use your admin account username and password
3. **Dashboard**: After login, you'll see the admin dashboard with system statistics

Admin Dashboard Features
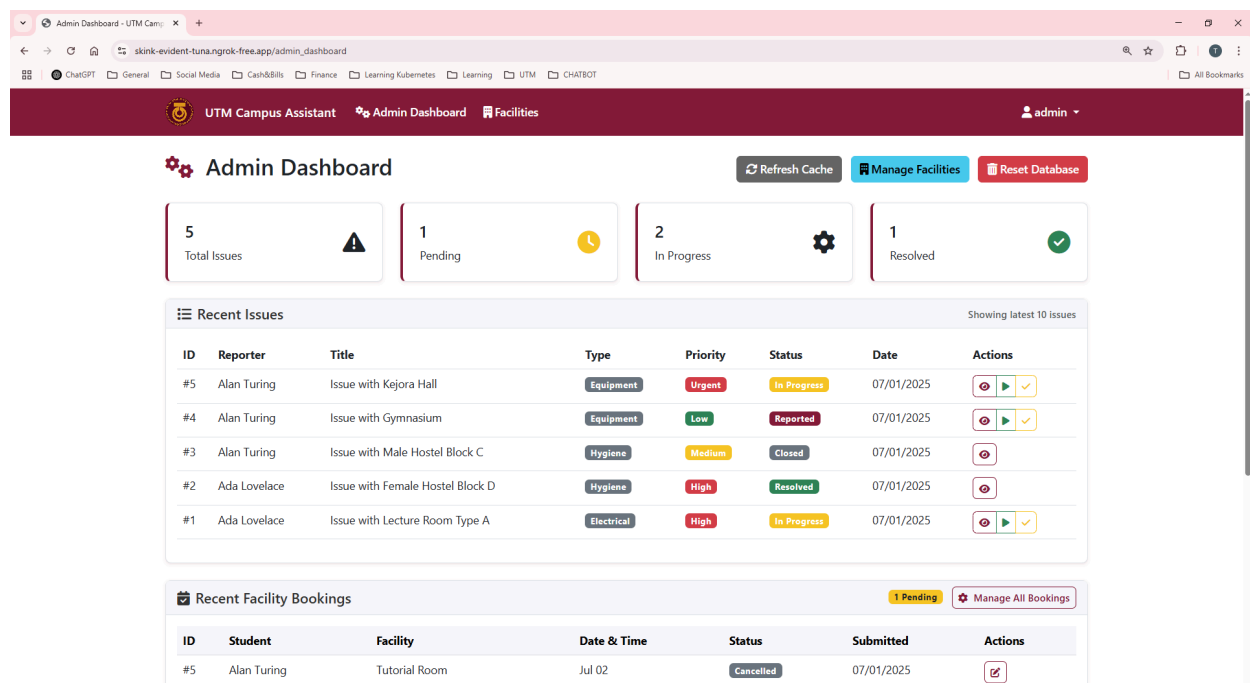
The dashboard provides:

- **System Statistics**: Active issues, bookings, user counts
- **Quick Actions**: Direct links to common tasks
- **Recent Activity**: Latest system events
- **Navigation Menu**: Access to all admin functions

First-Time Setup

1. **Create Admin Account**: Register with "Admin" role during initial setup
2. **Review Sample Data**: System includes sample facilities and test data
3. **Configure AI Service**: Ensure DeepSeek LLM is running (default: localhost:11434)
4. **Test Features**: Verify issue reporting, booking system, and chatbot functionality

Managing Issues

Issue Workflow

Issues follow this lifecycle:

```
REPORTED → IN_PROGRESS → RESOLVED → CLOSED
```

Viewing and Managing Issues

1. **Issues List**: Access via "Manage Issues" in admin dashboard
2. **Filtering**: Sort by status, priority, category, or assignment
3. **Issue Details**: Click any issue to view full information

Issue Classification System

Category	Examples	Typical Priority	Response Time
Electrical	Power outages, faulty outlets	HIGH/URGENT	2-4 hours
Hygiene	Cleaning, sanitation	MEDIUM	24 hours
Structural	Building damage, leaks	HIGH	4-8 hours
Equipment	Broken lab equipment	MEDIUM/HIGH	8-24 hours
Security	Access control, locks	HIGH/URGENT	1-2 hours
Other	General maintenance	LOW/MEDIUM	48 hours

Processing Issues

1. **Review**: Check AI-generated classification and priority
2. **Assign**: Assign to yourself or technician
3. **Update Status**: Move through workflow stages
4. **Add Notes**: Document actions taken and progress
5. **Resolve**: Mark complete when fixed
6. **Close**: Final closure after student feedback

Facility Management

Facility Information System

Adding New Facilities:

1. Navigate to "Manage Facilities"
2. Click "Add New Facility"
3. Complete required information:
   - Name and category
   - Location and capacity
   - Operating hours
   - Booking availability

Facility Categories:

- Laboratory (research and teaching labs)
- Academic (classrooms, lecture halls)
- Sports (gymnasiums, fields, courts)
- Administrative (offices, meeting rooms)

- Accommodation (dormitories)
- Dining (cafeterias, food courts)
- Event (auditoriums, conference centers)

Booking Management

Booking Approval Process:

1. **Review Requests**: Check pending booking requests
2. **Evaluate**: Verify availability, purpose, and user credentials
3. **Decision**: Approve, reject, or keep pending

Booking Actions:

- **APPROVED**: Confirm booking with optional notes
- **REJECTED**: Decline with reason for student
- **PENDING**: Keep under review
- **CANCELLED**: Handle user cancellations

Conflict Resolution

Common Conflicts:

- Time overlaps for same facility
- Capacity exceeded
- Maintenance during booking
- Policy violations

Resolution Strategies:

- Contact students for alternative times
- Suggest similar facilities
- Implement waiting lists
- Create priority systems

Troubleshooting

Common Issues and Solutions

Application Won't Start

```
# Check database connection
pg_isready -h localhost -p 5432

# Verify environment variables
echo $DATABASE_URL
echo $SESSION_SECRET

# Kill conflicting processes
pkill -f gunicorn
```

Database Issues

```
# Recreate tables if needed
python -c "from flask_app import app, db; app.app_context().push(); db.create_all
```

```
# Reset to clean state (admin interface)
# Access /admin/reset-database
```

AI Service Problems

```
# Check DeepSeek LLM service
curl http://localhost:11434/v1/models

# Restart Ollama service
pkill ollama
ollama serve
```

Login/Authentication Issues

- Check user exists in database
- Verify password hash format
- Clear browser cookies
- Check session configuration

Performance Issues

- Monitor memory usage: `htop`
- Check database query performance
- Verify AI service response times
- Optimize database connections

Emergency Procedures

Complete System Down:

1. Check service status: `systemctl status postgresql`
2. Restart services: `sudo systemctl restart postgresql`
3. Restart application: `gunicorn --bind 0.0.0.0:5000 main:app`

Data Recovery:

1. Stop application: `pkill gunicorn`
2. Restore database: `psql utm_campus_assistant < backup_latest.sql`
3. Verify restoration: Check table counts and user accounts
4. Restart application

Monitoring Checklist
```

**Daily Tasks:**

- Review new issue reports
- Process pending facility bookings
- Monitor system performance
- Check AI service health

**Weekly Tasks:**

- Analyze facility usage patterns
- Review user feedback
- Update facility information
- Generate utilization reports

**Monthly Tasks:**

- Database maintenance
- Security audit
- Backup verification
- Performance optimization

## Getting Help

**When to Escalate:**

- Security incidents
- Data corruption
- System performance severely degraded
- Multiple component failures
- Unable to resolve within 2 hours

**Support Information to Provide:**

- Problem description
- Steps already attempted
- Error messages and logs
- System configuration
- User impact assessment

# Quick Reference

## Key URLs

- **Main Application**: `http://localhost:5000`
- **Admin Dashboard**: `/admin/dashboard`
- **Issue Management**: `/admin/issues`
- **Booking Management**: `/admin/bookings`
- **Database Reset**: `/admin/reset-database`

## Important Files

- **Configuration**: `flask_app.py`
- **Database Models**: `models.py`
- **AI Service**: `ai_service.py`
- **Routes**: `routes.py`
- **Forms**: `forms.py`

## Environment Variables

```
DATABASE_URL="postgresql://user:pass@host:port/database"
SESSION_SECRET="your-secret-key"
DEEPSEEK_API_URL="http://localhost:11434/v1/chat/completions"
DEEPSEEK_MODEL="deepseek-r1:7b"
```

Default Admin Account

- **Username**: admin
- **Email**: admin@utm.edu.my
- **Role**: Admin
- **Note**: Set secure password during registration

This manual covers the essential administrative functions of the UTM Campus Assistant Chatbot. For detailed technical documentation, refer to the complete Sphinx documentation.