

HOSTELIX PRO

Product Features Document

Comprehensive Hostel Management Platform
for Educational Institutions

Version 1.0 | February 2026

Cross-Platform Mobile (Android & iOS) + RESTful Backend

Table of Contents

1. Executive Summary
2. System Architecture
3. User Roles & Permissions
4. Authentication & Security
5. Dashboard Module
6. User Management
7. Student Profiles
8. Fee Management
9. Routine Management
10. Daily Reports
11. Announcements & Holidays
12. Notifications
13. Audit Logs
14. Backup & Restore
15. Account Self-Service
16. Settings & Configuration
17. Data Export
18. Remote Access & Deployment
19. Technical Specifications

1. Executive Summary

Hostelix Pro is a comprehensive hostel management platform designed for educational institutions. It provides end-to-end digital management of student residences, including fee tracking, daily routine monitoring, activity reporting, and secure multi-role access control.

The system serves **four distinct user roles** — Administrators, Teachers, Routine Managers, and Students — each with tailored dashboards, permissions, and workflows. Built with a modern Flutter frontend and a Python Flask backend, Hostelix Pro delivers a seamless, real-time experience across Android and iOS devices.

Key Highlights

Capability	Description
Multi-Factor Authentication	Email OTP + optional TOTP 2FA via authenticator apps
Role-Based Access Control	Four roles with granular permissions per endpoint
Real-Time Dashboards	Role-specific statistics and activity feeds
Fee Lifecycle Management	Structure definition → Submission → Approval → PDF Challan
Routine Tracking	Walk/Exit requests with approval workflows and return monitoring
Encrypted Backups	Full database backup with AES encryption and key-based restore
Remote Access	Ngrok tunnel integration for access from any network

2. System Architecture

Hostelix Pro follows a **client-server architecture** with a Flutter mobile frontend communicating with a Flask RESTful API backend over HTTPS. The system supports both local development (SQLite) and production deployment (PostgreSQL).

Technology Stack

Layer	Technology	Purpose
Frontend	Flutter 3.x / Dart	Cross-platform mobile application
Backend	Python 3.x / Flask 3.0	RESTful API server
Database	SQLite (dev) / PostgreSQL (prod)	Persistent data storage
ORM	SQLAlchemy + Flask-Migrate	Database abstraction and migrations
Auth	PyJWT + bcrypt + PyOTP	JWT tokens, password hashing, TOTP
Email	SMTP (configurable)	OTP delivery and password reset
Export	ReportLab (PDF) / OpenPyXL (Excel)	Document generation
Tunnel	Ngrok	Secure remote access

3. User Roles & Permissions

Hostelix Pro implements a strict **Role-Based Access Control (RBAC)** system with four predefined roles, each with specific capabilities and restrictions.

3.1 Administrator (admin)

The top-level role with full system access:

- Create, update, delete, lock/unlock user accounts
- Approve pending student registrations
- Manage fee structures and approve/reject fee submissions
- View all reports, routines, and audit logs
- Create and manage announcements
- Create encrypted backups and perform restores
- Export data to Excel and PDF formats
- Access the full administrative dashboard

3.2 Teacher (teacher)

Responsible for overseeing assigned students:

- View profiles of assigned students only
- Approve or reject daily reports from assigned students
- Create announcements targeted at their students
- Access the teacher dashboard with daily student overview
- View routine history for assigned students

3.3 Routine Manager (routine_manager)

Manages student movement in and out of the hostel:

- Approve or reject walk and exit requests
- Confirm student returns to the hostel
- Track students currently out of the hostel
- Monitor late returns with time-based alerts
- Access the routine manager dashboard with live status

3.4 Student (student)

The end-user of the hostel system:

- Self-register with email (pending admin approval)

- Submit daily wake-up reports
- Create walk and exit requests with return confirmation
- Submit monthly fee payments with proof uploads
- View personal fee calendar, routines, and report history
- View announcements from admin, teacher, and managers
- Manage personal profile (bio, skills, status message)

4. Authentication & Security

4.1 Registration Flow

Students self-register with email, password, and display name. Accounts are created with **is_approved = false** and require explicit admin approval. During approval, the admin assigns room number, admission number, and a supervising teacher.

4.2 Multi-Factor Authentication (MFA) Login

Login follows a **two-step MFA process**:

- **Step 1:** Email + password verification → 6-digit OTP sent to registered email
- **Step 2:** OTP verification → JWT token issued with user claims
- OTP has a **5-minute expiry** window with max 5 attempts per transaction
- JWT tokens carry user ID, role, and expiration claims

4.3 Account Lockout Protection

Parameter	Value
Max failed login attempts	5
Lockout duration	30 minutes
Auto-unlock	Yes (time-based)
Admin manual lock/unlock	Yes

4.4 Two-Factor Authentication (TOTP)

Users can enable TOTP-based 2FA via authenticator apps (Google Authenticator, Authy). QR code generated server-side using *pyotp* and *qrcode*. Setup requires initial code verification. Disabling 2FA requires password confirmation. All 2FA actions are logged in the audit trail.

4.5 Password Reset Flow

- User submits email via forgot-password endpoint
- Backend sends 6-digit OTP to the email address
- User submits OTP + new password to reset
- OTP validated, password updated, lockout counters reset
- Branded HTML email template with security notices

5. Dashboard Module

Each user role has a tailored dashboard providing real-time insights and quick actions. Dashboards feature a cyberpunk-themed animated UI with dark mode optimization.

5.1 Admin Dashboard

Widget	Data Displayed
Total Users	Count of all registered users
Total Students	Count of active students
Fee Collection	Monthly collection summary and statistics
Pending Reports	Reports awaiting teacher/admin action
Active Routines	Currently active walk/exit requests
Recent Announcements	Latest system-wide announcements

5.2 Teacher Dashboard

Widget	Data Displayed
Assigned Students	List of students under supervision
Daily Status	Per-student wake-up report status for today
Pending Reports	Reports pending teacher approval
Quick Actions	Approve/reject reports inline

5.3 Routine Manager Dashboard

Widget	Data Displayed
Pending Requests	Walk/exit requests awaiting approval
Currently Out	Live count and list of students outside
Late Returns	Students past expected return time
Today's Activity	Summary of approved/rejected/completed routines

5.4 Student Dashboard

Widget	Data Displayed
Daily Report	Today's wake-up report status

Widget	Data Displayed
Active Routine	Current walk/exit status
Fee Summary	Outstanding and upcoming fee amounts
Quick Actions	Submit report, request walk/exit

6. User Management

Access: Admin only

6.1 User CRUD Operations

Operation	Endpoint	Description
List Users	GET /users	List all users with role and status filters
Get User	GET /users/{id}	Retrieve full user profile by ID
Create User	POST /users	Create user with email, password, role, name
Update User	PATCH /users/{id}	Update email, name, role, lock status
Delete User	DELETE /users/{id}	Permanently remove user and data
Lock/Unlock	POST /users/{id}/lock	Toggle account lock status
Approve	POST /users/{id}/approve	Approve pending registration

6.2 Student Approval Workflow

When approving a student, the admin assigns:

- **Admission Number** — Unique identifier within the institution
- **Room Number** — Hostel room assignment
- **Assigned Teacher** — Teacher responsible for daily reports

7. Student Profiles

7.1 Role-Based Access

Role	Access Level
Admin	All student profiles, full details
Teacher	Assigned students only
Routine Manager	All students, limited fields

7.2 Student Profile Fields

Field	Type	Description
admission_no	String	Unique admission number

Field	Type	Description
room	String	Assigned hostel room
assigned_teacher_id	Integer	Supervising teacher's user ID
profile_json	JSON	Extended profile data
bio	Text	Student bio / interests
skills	Text	Skills and competencies
status_message	String	Current status

7.3 Student Activity Calendar

Detailed activity history grouped by date, including reports, routines, and fee submissions. Admin can access via **GET /users/{id}/activities**.

8. Fee Management

8.1 Fee Structure Definition

Admins define reusable fee structures with name, amount, and description. Structures can be created, updated, and listed. Each student's monthly fee references a fee structure.

8.2 Fee Lifecycle

The fee payment follows a structured workflow:

- **UNPAID** → Fee record created for the month
- **PENDING_ADMIN** → Student submits payment with proof
- **APPROVED / PAID** → Admin approves the submission
- **REJECTED** → Admin rejects (with reason); student can resubmit
- **PARTIAL** → Partial payment submitted

8.3 Fee Calendar (Matrix View)

The fee calendar provides a **student x month matrix** showing fee status for an entire year. Each cell includes status, expected amount, paid amount, and pending amount. Admins see all students; students see only their own row. Yearly totals are calculated automatically.

8.4 Additional Fee Features

Feature	Description
Fee Statistics	Current month collection summary with outstanding balances
Proof Upload	Students upload payment proof images via multipart upload
Transaction History	All transactions per fee with status tracking
Fee Challan (PDF)	Professional PDF challan generated with ReportLab
Approve / Reject	Admin reviews and acts on individual transactions

9. Routine Management

9.1 Request Types

Type	Description	Who Creates
Walk	Short-duration outing within campus vicinity	Student
Exit	Extended departure requiring formal approval	Student
Return	Check-in confirmation after walk/exit	Student → Manager

9.2 Routine Lifecycle

The routine follows a multi-step approval workflow:

- **PENDING_ROUTINE_MANAGER** → Student submits walk/exit request
- **APPROVED_PENDING_RETURN** → Manager approves the request
- **PENDING_RETURN_APPROVAL** → Student requests return check-in
- **COMPLETED** → Manager confirms the return
- **REJECTED / RETURN_REJECTED** → Manager rejects (with reason)

9.3 Live Monitoring

Feature	Description
Currently Out	Real-time list of students outside the hostel
Late Returns	Students who exceeded expected return time
Pending Approvals	Requests awaiting manager action
Calendar View	Monthly view of routines grouped by date with daily summaries

10. Daily Reports

10.1 Report Flow

Students submit daily wake-up reports which go through a **multi-tier approval** process. Reports flow from Student → Teacher → Admin.

10.2 Report Status

Status	Description
PENDING_TEACHER	Awaiting assigned teacher's review
APPROVED	Teacher or admin approved the report
REJECTED	Teacher or admin rejected (reason required)

10.3 Role-Based Access

Role	Visible Reports
Student	Own reports only
Teacher	Reports from assigned students (filterable)
Admin	All reports (filterable by status)

Admin can export all reports to **Excel (.xlsx)** format.

11. Announcements & Holidays

11.1 Announcement Types & Priorities

Type	Priority Levels	Description
general	low / normal / high / urgent	Standard informational announcement
holiday	normal / high	Holiday declaration with date range
event	normal / high	Event notification with specific date
urgent	urgent	High-priority alert (top of feed)

11.2 Targeting & Visibility

Creator Role	Visibility Scope
Admin	All users (or targeted by role)
Routine Manager	All users (or targeted by role)
Teacher	Only their assigned students

Students only see announcements from admins, routine managers, and their assigned teacher. Holiday calendar endpoint returns all holiday announcements for a given year.

12. Notifications

Feature	Description
Per-User Inbox	Each user has a personal notification feed
Read/Unread State	Notifications track individual read status
Badge Count	Unread count via dedicated endpoint
Mark as Read	Individual or bulk mark-all-as-read support
Filtering	Filter by unread status with configurable limits

13. Audit Logs

Access: Admin only

Every sensitive action in the system is logged with comprehensive metadata.

Action Category	Examples
Authentication	LOGIN_SUCCESS, LOGIN_FAILED, LOGOUT
User Management	USER_CREATE, USER_UPDATE, USER_DELETE, USER_LOCK
Profile Changes	UPDATE_PROFILE, CHANGE_PASSWORD
Security	ENABLE_2FA, DISABLE_2FA
Fee Operations	FEE_APPROVE, FEE_REJECT
Announcements	ANNOUNCEMENT_CREATE, ANNOUNCEMENT_DELETE
Backups	BACKUP_CREATE, BACKUP_RESTORE_VERIFY

Each entry captures: User ID, Action, Entity, Entity ID, IP Address, User-Agent, Details (JSON), and Timestamp. Results are filterable and paginated.

14. Backup & Restore

Access: Admin only

Operation	Description
Create Backup	Full database export with AES encryption; returns unique key
List Backups	View all backups with metadata (size, timestamp)
Download Backup	Download encrypted backup file
Restore Backup	Verify and restore using the original encryption key

Backup files are encrypted at rest. Restoration requires the exact encryption key provided during creation. All operations are logged in the audit trail.

15. Account Self-Service

Feature	Details
Profile Edit	Display name, email, bio, skills, status message
Password Change	Requires current password; min 6 characters
2FA Setup	Generate QR code + secret; verify with authenticator code
2FA Disable	Requires current password confirmation

16. Settings & Configuration

Setting	Description
Theme Mode	Dark (default) / Light mode toggle with persistence
Backend URL	Configurable API host for remote access
URL Cleanup	Auto-strips trailing slashes and /api/v1 suffixes
Reset Option	One-tap reset to default host URL

17. Data Export

Data Type	Format	Access	Technology
Reports	.xlsx (Excel)	Admin	OpenPyXL

Data Type	Format	Access	Technology
Fee Challans	.pdf	Admin / Student	ReportLab

18. Remote Access & Deployment

18.1 Ngrok Integration

- Automated setup via **run_remote.bat** script
- Auto-detects ngrok binary (PATH or local directory)
- Configures auth token and launches tunnel automatically
- **ngrok-skip-browser-warning** header added to all API requests
- CORS configured for all origins

18.2 Deployment Options

Platform	Type	Complexity
Ngrok	Quick tunnel (dev/testing)	Low
Local Network	Direct IP on shared WiFi	Low
Render	Cloud PaaS (production)	Medium
Railway	Cloud PaaS (production)	Medium
Fly.io	Edge deployment (production)	Medium

19. Technical Specifications

19.1 API Endpoint Summary

Module	Base Path	Endpoints
Auth	/api/v1/auth	7
Users	/api/v1/users	9
Account	/api/v1/account	6
Dashboard	/api/v1/dashboard	3
Reports	/api/v1/reports	5
Routines	/api/v1/routines	8
Fees	/api/v1/fees	10
Announcements	/api/v1/announcements	4
Notifications	/api/v1/notifications	4

Module	Base Path	Endpoints
Audit	/api/v1/audit	1
Backups	/api/v1/backups	4
Total		61

19.2 Data Models

Model	Table	Purpose
User	users	Authentication and RBAC
Student	students	Extended student profile
Fee	fees	Monthly fee records
FeeStructure	fee_structures	Fee templates
Transaction	transactions	Payment transactions
Report	reports	Daily wake-up reports
Routine	routines	Walk/exit/return requests
Announcement	announcements	System announcements
Notification	notifications	User notification inbox
AuditLog	audit_logs	Security audit trail
BackupMeta	backup_meta	Backup file metadata

19.3 Frontend Page Map

Directory	Pages	Description
pages/auth/	6	Login, Signup, OTP, Forgot/Reset Password, Splash
pages/admin/	6	Dashboard, Users, Fees, Audit, Backup, Activities
pages/common/	4	Announcements, Reports, Student Profiles, Settings
pages/student/	3	Dashboard, Fee Page, Routine Page
pages/teacher/	2	Dashboard, My Students
pages/manager/	1	Routine Manager Dashboard
Total	22	

End of Document — Hostelix Pro v1.0 Product Features