Cricket Application — Complete Folder Hierarchy

📁 Full Project Structure

```
cricket_app/
│
├── .env                          # Environment variables (secrets, DB URLs) — NEVER commit to git
├── .gitignore                    # Files to ignore in git
├── requirements.txt              # All pip packages needed
├── run.py                        # Entry point — starts the Flask app
│
├── app/                          # Main application package
│   │
│   ├── __init__.py               # App factory — creates and configures Flask app
│   │
│   ├── config.py                 # All configuration (DB URI, SECRET_KEY, Redis, etc.)
│   │
│   ├── extensions.py             # Third-party extensions (db, socketio, cache, login_manager)
│   │
│   ├── models/                   # All database models (tables)
│   │   ├── __init__.py           # Imports all models so Flask finds them
│   │   ├── team.py               # Team model
│   │   ├── player.py             # Player model
│   │   ├── match.py              # Match model
│   │   ├── innings.py            # Innings model
│   │   ├── ball.py               # Ball model
│   │   ├── partnership.py        # Partnership model
│   │   ├── scorecard.py          # BattingScorecard & BowlingScorecard models
│   │   ├── commentary.py         # Commentary model
│   │   ├── highlight.py          # Highlight model
│   │   ├── tournament.py         # Tournament & TournamentTeam models
│   │   └── user.py               # User model (auth)
│   │
│   ├── routes/                   # All API and page routes
│   │   ├── __init__.py           # Registers all route blueprints
│   │   ├── main.py               # Home page, match pages (renders templates)
│   │   ├── api/                  # All REST API endpoints
│   │   │   ├── __init__.py       # API blueprint setup
│   │   │   ├── matches.py        # /api/v1/matches — CRUD for matches
│   │   │   ├── teams.py          # /api/v1/teams — CRUD for teams
```

```
│   │   │   ├── players.py          # /api/v1/players — CRUD for players
│   │   │   ├── innings.py          # /api/v1/innings — start/manage innings
│   │   │   ├── balls.py            # /api/v1/balls — record deliveries
│   │   │   ├── stats.py            # /api/v1/stats — player & team statistics
│   │   │   ├── prediction.py       # /api/v1/predict — match predictions
│   │   │   └── auth.py             # /api/v1/auth — login, register, token refresh
│   │   └── pages.py                # Rendered HTML pages (dashboard, match view, etc.)
│   │
│   ├── services/                   # Business logic layer — does the heavy work
│   │   ├── __init__.py
│   │   ├── match_service.py        # Create match, get summary, update status
│   │   ├── ball_service.py         # Record ball, handle over/striker rotation
│   │   ├── innings_service.py      # Start innings, check completion, set target
│   │   ├── statistics_service.py   # Batting stats, bowling stats, scorecards
│   │   ├── prediction_service.py   # Team strength, win probability, explanations
│   │   ├── commentary_service.py   # Auto-generate commentary from ball data
│   │   └── tournament_service.py   # Points table, scheduling, standings
│   │
│   ├── validators/                 # Input validation schemas (Marshmallow)
│   │   ├── __init__.py
│   │   ├── match_validator.py      # Validate match creation payload
│   │   ├── ball_validator.py       # Validate ball recording payload
│   │   ├── team_validator.py       # Validate team creation payload
│   │   ├── player_validator.py     # Validate player creation payload
│   │   └── auth_validator.py       # Validate login/register payload
│   │
│   ├── middleware/                 # Request/response middleware
│   │   ├── __init__.py
│   │   ├── auth.py                 # JWT token verification (@token_required)
│   │   ├── rate_limiter.py         # Rate limiting (prevent API abuse)
│   │   └── error_handler.py        # Global error handling (404, 500, etc.)
│   │
│   ├── websockets/                 # Real-time communication
│   │   ├── __init__.py
│   │   └── match_socket.py         # Live score updates via SocketIO
│   │
│   ├── templates/                  # HTML templates (Jinja2)
│   │   ├── base.html               # Base layout — header, nav, footer (all pages extend this)
│   │   ├── home.html               # Dashboard / home page
│   │   ├── match.html              # Single match scorecard view
│   │   ├── player_stats.html       # Player statistics page
│   │   ├── prediction.html         # Prediction results page
│   │   └── auth/
│   │       ├── login.html          # Login page
```

```
│   │   └── register.html      # Register page
│   │
│   └── static/                # Front-end assets
│       ├── css/
│       │   └── style.css      # Custom styles
│       ├── js/
│       │   ├── main.js        # General app logic
│       │   ├── scorecard.js   # Live scorecard updates (SocketIO client)
│       │   └── charts.js      # Chart rendering logic
│       └── img/
│           └── logos/         # Team logos, app logo, etc.
│
├── migrations/                # Database migration files (Flask-Migrate)
│   ├── __init__.py
│   ├── env.py                 # Migration environment config
│   └── versions/              # Each migration is a versioned file
│       └── 0001_initial.py    # First migration — creates all tables
│
├── tests/                     # All test files
│   ├── __init__.py
│   ├── conftest.py            # Shared test fixtures (test DB, test client, etc.)
│   ├── test_models/
│   │   ├── __init__.py
│   │   ├── test_team.py       # Tests for Team model
│   │   ├── test_player.py     # Tests for Player model
│   │   ├── test_match.py      # Tests for Match model
│   │   └── test_ball.py       # Tests for Ball model
│   ├── test_services/
│   │   ├── __init__.py
│   │   ├── test_match_service.py  # Tests for match business logic
│   │   ├── test_ball_service.py   # Tests for ball recording logic
│   │   └── test_prediction.py     # Tests for prediction logic
│   └── test_routes/
│       ├── __init__.py
│       ├── test_match_api.py      # Tests for match API endpoints
│       ├── test_ball_api.py       # Tests for ball API endpoints
│       └── test_auth_api.py       # Tests for auth endpoints
│
├── scripts/                   # Utility scripts
│   ├── seed_data.py           # Populate DB with sample teams/players
│   ├── clear_db.py            # Wipe all data (your current cleardb.py)
│   └── backup_db.py           # Backup the database
│
└── docker/                    # Docker configuration (for deployment)
```

```
├── Dockerfile              # Builds the app container
├── docker-compose.yml      # Runs app + Redis + DB together
└── nginx.conf              # Reverse proxy config
```

🔗 How the Folders Connect

```
run.py
  │
  ▼ imports and starts
app/__init__.py  (App Factory)
  │
  ├── loads config.py          (all settings)
  ├── initializes extensions.py (db, cache, socketio)
  ├── registers routes/        (all blueprints/endpoints)
  │     ├── main.py            (page routes)
  │     └── api/               (REST API routes)
  │           ├── matches.py
  │           ├── balls.py     ──► calls ──► services/ball_service.py
  │           ├── stats.py     ──► calls ──► services/statistics_service.py
  │           └── ...
  ├── registers middleware/    (auth, errors, rate limiting)
  └── registers websockets/    (live updates)

      services/ (business logic)
        │
        ├── reads/writes ──► models/ (database tables)
        └── validates via ──► validators/ (input checks)
```

📄 What Goes Inside Each Key File

`run.py` — The Starting Point

```python
```

```python
from app import create_app

app = create_app()

if __name__ == "__main__":
    app.run(debug=True)
```

`app/__init__.py` — The App Factory

```python
from flask import Flask
from .extensions import db, socketio, cache, login_manager
from .config import Config

def create_app():
    app = Flask(__name__)
    app.config.from_object(Config)

    # Initialize extensions
    db.init_app(app)
    socketio.init_app(app)
    cache.init_app(app)
    login_manager.init_app(app)

    # Register blueprints (routes)
    from .routes import register_blueprints
    register_blueprints(app)

    # Register error handlers
    from .middleware.error_handler import register_errors
    register_errors(app)

    # Create tables
    with app.app_context():
        db.create_all()

    return app
```

`app/extensions.py` — All Third-Party Tools

```python
```

```
from flask_sqlalchemy import SQLAlchemy
from flask_socketio import SocketIO
from flask_caching import Cache
from flask_login import LoginManager

db = SQLAlchemy()
socketio = SocketIO()
cache = Cache()
login_manager = LoginManager()
```

`app/models/__init__.py` — Import All Models

```
python

# This file makes sure Flask knows about every model
from .team import Team
from .player import Player
from .match import Match
from .innings import Innings
from .ball import Ball
from .partnership import Partnership
from .scorecard import BattingScorecard, BowlingScorecard
from .commentary import Commentary
from .highlight import Highlight
from .tournament import Tournament, TournamentTeam
from .user import User
```

`app/routes/__init__.py` — Register All Blueprints

```
python

from .main import main_bp
from .pages import pages_bp
from .api import api_bp

def register_blueprints(app):
    app.register_blueprint(main_bp)
    app.register_blueprint(pages_bp)
    app.register_blueprint(api_bp, url_prefix='/api/v1')
```

`app/routes/api/__init__.py` — API Blueprint Setup

```python
from flask import Blueprint

api_bp = Blueprint('api', __name__)

# Import all API route files so they register
from . import matches, teams, players, innings, balls, stats, prediction, auth
```

`app/config.py` — All Settings in One Place

```python
import os
from dotenv import load_dotenv

load_dotenv()  # reads .env file

class Config:
    SECRET_KEY = os.environ.get('SECRET_KEY', 'fallback-secret')
    SQLALCHEMY_DATABASE_URI = os.environ.get('DATABASE_URL', 'sqlite:///cricket.db')
    SQLALCHEMY_TRACK_MODIFICATIONS = False

    # Redis (for caching)
    CACHE_TYPE = 'redis'
    CACHE_REDIS_URL = os.environ.get('REDIS_URL', 'redis://localhost:6379/0')

    # SocketIO
    SOCKETIO_ASYNC_MODE = 'threading'
```

`.env` — Secrets (Never Commit This)

```
SECRET_KEY=your-super-secret-key-here
DATABASE_URL=sqlite:///cricket.db
REDIS_URL=redis://localhost:6379/0
```

`requirements.txt` — All Packages

```
Flask
Flask-SQLAlchemy
```

```
Flask-Migrate
Flask-SocketIO
Flask-Caching
Flask-Login
marshmallow
PyJWT
python-dotenv
redis
gunicorn
```

🗂️ Your Current Files → Where They Move

Your Current File	New Location	Reason
`__init__.py`	`app/__init__.py`	App factory stays here
`config.py`	`app/config.py`	Inside the app package
`extensions.py`	`app/extensions.py`	Inside the app package
`models.py`	`app/models/*.py`	Split into one file per model
`routes.py`	`app/routes/api/*.py`	Split by resource (matches, balls, etc.)
`services.py`	`app/services/*.py`	Split by responsibility
`cleardb.py`	`scripts/clear_db.py`	Utility scripts go in scripts/
`templates/`	`app/templates/`	Inside the app package
`static/`	`app/static/`	Inside the app package