Task Tracker API - Notes (Roman Urdu)

Project Setup

Step 1: Project Folder aur Virtual Environment

1. Terminal mein jao:

```
cd path/to/task-tracker
```

2. Virtual environment banao:

```
python -m venv env
```

3. Activate karo: **Windows:**

```
env\Scripts\activate
```

Linux/Mac:

```
source env/bin/activate
```

4. Dependencies install karo:

```
pip install fastapi uvicorn pydantic
```

Step 2: API Server Run Karo

```
uvicorn main:app --reload
```

Browser mein `http://127.0.0.1:8000/docs` kholo.

Project Files

File	Kaam
main.py	FastAPI app aur endpoints
models.py	In-memory data store
schemas.py	Pydantic models for validation

models.py

```python
from typing import List
from schemas import UserRead, Task

users_db: List[UserRead] = []
tasks_db: List[Task] = []
```

Note: Temporary store. Production ke liye SQLite/PostgreSQL use karo.

schemas.py

```python
from pydantic import BaseModel, EmailStr, constr, field_validator
from datetime import date
from typing import Optional, Annotated
```

Models

UserCreate

```python
class UserCreate(BaseModel):
    username: Annotated[str, constr(min_length=3, max_length=20)]
    email: EmailStr
```

UserRead

```python
class UserRead(BaseModel):
    id: int
    username: str
    email: EmailStr
```

TaskBase

```python
class TaskBase(BaseModel):
    title: str
    description: Optional[str] = None
    due_date: date
    status: str = "pending"

    @field_validator("due_date")
    @classmethod
    def due_date_cannot_be_in_past(cls, v):
        if v < date.today():
            raise ValueError("Due␣date␣cannot␣be␣in␣the␣past")
        return v

    @field_validator("status")
```

```
    @classmethod
    def validate_status(cls, v):
        allowed_statuses = {"pending", "in_progress", "completed"}
        if v not in allowed_statuses:
            raise ValueError(f"Status must be one of: {allowed_statuses}")
        return v
```

TaskCreate

```
class TaskCreate(TaskBase):
    user_id: int
```

Task

```
class Task(TaskBase):
    id: int
    user_id: int
```

main.py

Imports

```
from fastapi import FastAPI, HTTPException
from models import users_db, tasks_db
from schemas import UserCreate, UserRead, TaskCreate, Task
```

App

```
app = FastAPI()
```

Root Endpoint

```
@app.get("/")
def read_root():
    return {"message": "Task Tracker API"}
```

User Endpoints

Create User

```python
@app.post("/users/", response_model=UserRead)
def create_user(user: UserCreate):
    new_user = UserRead(id=len(users_db)+1, **user.dict())
    users_db.append(new_user)
    return new_user
```

Get User

```python
@app.get("/users/{user_id}", response_model=UserRead)
def get_user(user_id: int):
    for user in users_db:
        if user.id == user_id:
            return user
    raise HTTPException(status_code=404, detail="User not found")
```

Task Endpoints

Create Task

```python
@app.post("/tasks/", response_model=Task)
def create_task(task: TaskCreate):
    new_task = Task(id=len(tasks_db)+1, **task.dict())
    tasks_db.append(new_task)
    return new_task
```

Get Task

```python
@app.get("/tasks/{task_id}", response_model=Task)
def get_task(task_id: int):
    for task in tasks_db:
        if task.id == task_id:
            return task
    raise HTTPException(status_code=404, detail="Task not found")
```

Update Task Status

```python
@app.put("/tasks/{task_id}", response_model=Task)
def update_task_status(task_id: int, status: str):
    allowed_statuses = {"pending", "in_progress", "completed"}
    if status not in allowed_statuses:
        raise HTTPException(status_code=400, detail=f"Status must
            be one of: {allowed_statuses}")
    for task in tasks_db:
        if task.id == task_id:
            task.status = status
            return task
    raise HTTPException(status_code=404, detail="Task not found")
```

Get User Tasks

```python
@app.get("/users/{user_id}/tasks", response_model=list[Task])
def get_user_tasks(user_id: int):
    return [task for task in tasks_db if task.user_id == user_id]
```

Common Errors

Error	Wajah	Fix
404 Not Found	Galat URL ya ID	/docs ya valid ID use karo
405 Method Not Allowed	Galat HTTP method	Decorator method check karo
Validation Error	Invalid due$_{d}$ateyastatus	JSON aur allowed values check karo

Improvements

1. Status update ke liye Pydantic model.
2. Username alphanumeric validation:

```python
constr(min_length=3, max_length=20, pattern=r"^[a-zA-Z0-9]+$"
    )
```

References

- FastAPI: `https://fastapi.tiangolo.com/`
- Pydantic: `https://docs.pydantic.dev/`
- HTTP Status: `https://developer.mozilla.org/en-US/docs/Web/HTTP/Status`