Sign in

Samples - Python (fastapi)

https://g.co/gemini/share/5d6fa2cd7b82

Created with **2.5 Pro (preview)** May 17, 2025 at 05:30 PM • Published May 17, 2025 at 05:43 PM

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 2389b4a..d1a42b7 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -5,8 +5,11 @@ from sqlmodel import Field, Relationship, SQLModel

 # Shared properties
-class UserBase(SQLModel):
+class EUserBase:
     email: EmailStr = Field(unique=True, index=True, max_length=255)
+
+
+class UserBase(SQLModel, EUserBase):
     is_active: bool = True
     is_superuser: bool = False

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
    full_name: str | None = Field(default=None,
max_length=255)
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Extract Superclass

Justification:

- **Extract Superclass**: A new class, `EUserBase`, was created. The field `email` (along with its type `EmailStr` and `Field` attributes) was moved from the existing class `UserBase` to this new `EUserBase` class. Subsequently, the `UserBase` class was modified to inherit from `EUserBase` (in addition to `SQLModel`), effectively making `EUserBase` a new superclass for `UserBase` and encapsulating the `email` attribute.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

This pattern is used to share common attributes/behavior by placing them in a parent class.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**

```
diff --git backend/app/models.py backend/app/models.py
index d1a42b7..121e268 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -6,13 +6,13 @@ from sqlmodel import Field, Relationship, SQLModel

 # Shared properties
 class EUserBase:
+    full_name: str | None = Field(default=None, max_length=255)
     email: EmailStr = Field(unique=True, index=True, max_length=255)


 class UserBase(SQLModel, EUserBase):
     is_active: bool = True
     is_superuser: bool = False
-    full_name: str | None = Field(default=None, max_length=255)


 # Properties to receive via API on creation
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Pull Up Field

Justification:

- **Pull Up Field**: The field `full_name` was moved from the child class `UserBase` to its parent class `EUserBase`. This is shown by the removal of `full_name` from `UserBase` and its addition to `EUserBase`, which is established as a superclass of `UserBase` (as indicated by `class UserBase(SQLModel, EUserBase):`).

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 121e268..2481e66 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -7,10 +7,10 @@ from sqlmodel import Field, Relationship, SQLModel
 # Shared properties
 class EUserBase:
     full_name: str | None = Field(default=None, max_length=255)
-    email: EmailStr = Field(unique=True, index=True, max_length=255)

 class UserBase(SQLModel, EUserBase):
+   email: EmailStr = Field(unique=True, index=True, max_length=255)
     is_active: bool = True
     is_superuser: bool = False

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Push Down Field

Justification:

- **Push Down Field**: The field `email` was moved from the parent class `EUserBase` to its child class `UserBase`. This is indicated by the removal of the `email` field from `EUserBase` and its addition to `UserBase`, given that `UserBase` inherits from `EUserBase` (as shown by `class UserBase(SQLModel, EUserBase):`).

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 2481e66..65ca473 100644
--- backend/app/models.py
+++ backend/app/models.py

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
@@ -9,9 +9,12 @@ class EUserBase:
    full_name: str | None = Field(default=None,
max_length=255)


-class UserBase(SQLModel, EUserBase):
-    email: EmailStr = Field(unique=True, index=True,
max_length=255)
+class E2UserBase:
    is_active: bool = True
+
+
+class UserBase(SQLModel, EUserBase, E2UserBase):
+    email: EmailStr = Field(unique=True, index=True,
max_length=255)
    is_superuser: bool = False
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Extract Superclass

Justification:

- **Extract Superclass**: A new class, `E2UserBase`, was created. The field `is_active` (which was previously defined in `UserBase`) was moved into this new `E2UserBase` class. Subsequently, the `UserBase` class was modified to inherit from `E2UserBase` (in addition to its existing parents `SQLModel` and `EUserBase`). This makes `E2UserBase` a new superclass for `UserBase`, encapsulating the `is_active` attribute.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 65ca473..4a299e0 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -49,7 +49,7 @@ class UpdatePassword(SQLModel):
 class User(UserBase, table=True):
     id: uuid.UUID = Field(default_factory=uuid.uuid4, primary_key=True)
     hashed_password: str
-    items: list["Item"] = Relationship(back_populates="owner", cascade_delete=True)

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
+    items: list["IItem"] =
Relationship(back_populates="owner", cascade_delete=True)


 # Properties to return via API, id is always required
@@ -79,7 +79,7 @@ class ItemUpdate(ItemBase):


 # Database model, database table inferred from class name
-class Item(ItemBase, table=True):
+class IItem(ItemBase, table=True):
     id: uuid.UUID = Field(default_factory=uuid.uuid4,
primary_key=True)
     owner_id: uuid.UUID = Field(
         foreign_key="user.id", nullable=False,
ondelete="CASCADE"
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

Do not generate explanations unrelated to the given transformation.

- Rename Class

Justification:

- **Rename Class**: The class `Item` has been renamed to `IItem`. This is evident from the change in the class definition (`-class Item(ItemBase, table=True):` to `+class IItem(ItemBase, table=True):`). The internal structure of the class (inheriting from `ItemBase` and its fields) remains the same. Consequently, the type hint for the `items` field in the `User` class was updated from `list["Item"]` to `list["IItem"]` to reflect this change.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/api/routes/login.py backend/app/api/routes/login.py
index 980c66f..9860460 100644
--- backend/app/api/routes/login.py
+++ backend/app/api/routes/login.py
@@ -91,7 +91,7 @@ def reset_password(session: SessionDep, body: NewPassword) -> Message:
)
    elif not user.is_active:
        raise HTTPException(status_code=400, detail="Inactive user")

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
-    hashed_password =
get_password_hash(password=body.new_password)
+    hashed_password =
get_password_hash(password=body.new_password_is)
     user.hashed_password = hashed_password
     session.add(user)
     session.commit()
diff --git backend/app/api/routes/users.py
backend/app/api/routes/users.py
index 6429818..481ce2b 100644
--- backend/app/api/routes/users.py
+++ backend/app/api/routes/users.py
@@ -106,11 +106,11 @@ def update_password_me(
     """
     if not verify_password(body.current_password,
current_user.hashed_password):
         raise HTTPException(status_code=400, detail="Incorrect
password")
-    if body.current_password == body.new_password:
+    if body.current_password == body.new_password_is:
         raise HTTPException(
             status_code=400, detail="New password cannot be
the same as the current one"
         )
-    hashed_password =
get_password_hash(body.new_password)
+    hashed_password =
get_password_hash(body.new_password_is)
     current_user.hashed_password = hashed_password
     session.add(current_user)
     session.commit()
diff --git backend/app/models.py backend/app/models.py
index 4a299e0..5ec1f06 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -116,4 +116,4 @@ class TokenPayload(SQLModel):
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
 class NewPassword(SQLModel):
     token: str
-    new_password: str = Field(min_length=8, max_length=40)
+    new_password_is: str = Field(min_length=8,
max_length=40)
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Rename Field

Justification:

- **Rename Field:** The field `new_password` in the class `NewPassword` (defined in `backend/app/models.py`) was renamed to `new_password_is`. This is evident from the

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

change in the `NewPassword` class definition:

Diff

```
--- backend/app/models.py
+++ backend/app/models.py
@@ -116,4 +116,4 @@ class TokenPayload(SQLModel):

 class NewPassword(SQLModel):
     token: str
-    new_password: str = Field(min_length=8, max_length=40)
+    new_password_is: str = Field(min_length=8, max_length=40)
```

Consequently, all usages of this field were updated in other files. For example, in `backend/app/api/routes/login.py` and `backend/app/api/routes/users.py`, references like `body.new_password` were changed to `body.new_password_is`.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/core/config.py
backend/app/core/config.py
index d58e03c..fb011b1 100644
--- backend/app/core/config.py
+++ backend/app/core/config.py
@@ -95,24 +95,35 @@ class Settings(BaseSettings):
    FIRST_SUPERUSER: EmailStr
    FIRST_SUPERUSER_PASSWORD: str

-    def _check_default_secret(self, var_name: str, value: str | None) -> None:
-        if value == "changethis":

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
+    @model_validator(mode="after")
+    def _enforce_non_default_secrets(self) -> Self:
+        if self.SECRET_KEY == "changethis":
             message = (
-                f'The value of {var_name} is "changethis", '
+                f'The value of {"SECRET_KEY"} is "changethis", '
                 "for security, please change it, at least for
deployments."
             )
             if self.ENVIRONMENT == "local":
                 warnings.warn(message, stacklevel=1)
             else:
                 raise ValueError(message)
-
-    @model_validator(mode="after")
-    def _enforce_non_default_secrets(self) -> Self:
-        self._check_default_secret("SECRET_KEY",
self.SECRET_KEY)
-        self._check_default_secret("POSTGRES_PASSWORD",
self.POSTGRES_PASSWORD)
-        self._check_default_secret(
-            "FIRST_SUPERUSER_PASSWORD",
self.FIRST_SUPERUSER_PASSWORD
-        )
+        if self.POSTGRES_PASSWORD == "changethis":
+            message1 = (
+                f'The value of {"POSTGRES_PASSWORD"} is
"changethis", '
+                "for security, please change it, at least for
deployments."
+            )
+            if self.ENVIRONMENT == "local":
+                warnings.warn(message1, stacklevel=1)
+            else:
+                raise ValueError(message1)
+        if self.FIRST_SUPERUSER_PASSWORD == "changethis":
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
+            message2 = (
+                f'The value of {"FIRST_SUPERUSER_PASSWORD"} is
"changethis", '
+                "for security, please change it, at least for
deployments."
+            )
+            if self.ENVIRONMENT == "local":
+                warnings.warn(message2, stacklevel=1)
+            else:
+                raise ValueError(message2)

        return self
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

- Inline Method

Justification:

- **Inline Method**: The method `_check_default_secret(self, var_name: str, value: str | None)` has been removed. Its functionality, which involved checking if a given configuration value was "changethis" and issuing a warning or error, has been directly incorporated (inlined) into its caller method, `_enforce_non_default_secrets`. The three distinct calls to `_check_default_secret` within `_enforce_non_default_secrets` (for `SECRET_KEY`, `POSTGRES_PASSWORD`, and `FIRST_SUPERUSER_PASSWORD`) have been replaced by their corresponding logic from the body of the former `_check_default_secret` method, with the `var_name` parameter effectively substituted with the specific string literal for each key.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 5ec1f06..080e548 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -15,11 +15,11 @@ class E2UserBase:

 class UserBase(SQLModel, EUserBase, E2UserBase):
     email: EmailStr = Field(unique=True, index=True, max_length=255)
-    is_superuser: bool = False

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
 # Properties to receive via API on creation
 class UserCreate(UserBase):
+    is_superuser: bool = False
     password: str = Field(min_length=8, max_length=40)


@@ -31,6 +31,7 @@ class UserRegister(SQLModel):

 # Properties to receive via API on update, all are optional
 class UserUpdate(UserBase):
+    is_superuser: bool = False
     email: EmailStr | None = Field(default=None,
max_length=255)  # type: ignore
     password: str | None = Field(default=None, min_length=8,
max_length=40)

@@ -47,6 +48,7 @@ class UpdatePassword(SQLModel):

 # Database model, database table inferred from class name
 class User(UserBase, table=True):
+    is_superuser: bool = False
     id: uuid.UUID = Field(default_factory=uuid.uuid4,
primary_key=True)
     hashed_password: str
     items: list["Item"] = Relationship(back_populates="owner",
cascade_delete=True)
@@ -54,6 +56,7 @@ class User(UserBase, table=True):

 # Properties to return via API, id is always required
 class UserPublic(UserBase):
+    is_superuser: bool = False
     id: uuid.UUID
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- **Push Down Field**

Justification:

- **Push Down Field**: The field `is_superuser: bool = False` was moved from the parent class `UserBase` to several of its child classes: `UserCreate`, `UserUpdate`, `User`, and `UserPublic`. This is shown by the removal of the `is_superuser` field from the `UserBase` class and its subsequent addition to each of these subclasses, which all inherit from `UserBase`.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 268e544..93f188a 100644
--- backend/app/models.py
+++ backend/app/models.py
@@ -14,13 +14,13 @@ class E2UserBase:

```
 class UserBase(SQLModel, EUserBase, E2UserBase):
+    password: str = Field(min_length=8, max_length=40)
     email: EmailStr = Field(unique=True, index=True,
max_length=255)


 # Properties to receive via API on creation
 class UserCreate(UserBase):
     is_superuser: bool = False
-    password: str = Field(min_length=8, max_length=40)


 class UserRegister(SQLModel):
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Pull Up Field

Justification:

- **Pull Up Field**: The field `password: str = Field(min_length=8, max_length=40)` was moved from the child class `UserCreate` to its parent class `UserBase`. This is evident as the `password` field was removed from `UserCreate` and added to `UserBase`, with `UserCreate` being a subclass of `UserBase`.

You are an expert coding assistant specialized in software refactoring, with many years of experience analyzing code transformations.

You will be given the diffs of a commit:

- **Diffs:**
diff --git backend/app/models.py backend/app/models.py
index 080e548..268e544 100644

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
--- backend/app/models.py
+++ backend/app/models.py
@@ -67,7 +67,7 @@ class UsersPublic(SQLModel):

 # Shared properties
 class ItemBase(SQLModel):
-    title: str = Field(min_length=1, max_length=255)
+    title_item: str = Field(min_length=1, max_length=255)
     description: str | None = Field(default=None,
max_length=255)


@@ -78,7 +78,7 @@ class ItemCreate(ItemBase):

 # Properties to receive on item update
 class ItemUpdate(ItemBase):
-    title: str | None = Field(default=None, min_length=1,
max_length=255)  # type: ignore
+    title_item: str | None = Field(default=None, min_length=1,
max_length=255)  # type: ignore


 # Database model, database table inferred from class name
diff --git backend/app/tests/api/routes/test_items.py
backend/app/tests/api/routes/test_items.py
index c215238..27f3c16 100644
--- backend/app/tests/api/routes/test_items.py
+++ backend/app/tests/api/routes/test_items.py
@@ -34,7 +34,7 @@ def test_read_item(
     )
     assert response.status_code == 200
     content = response.json()
-    assert content["title"] == item.title
+    assert content["title"] == item.title_item
     assert content["description"] == item.description
     assert content["id"] == str(item.id)
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
    assert content["owner_id"] == str(item.owner_id)
```

Your task is to identify which refactoring type(s) have been applied in transforming the original program into the new version. Use only the following list of predefined refactorings:

(Add Method Parameter) – Introduces a new parameter to an existing method.
(Remove Method Parameter) – Eliminates an existing parameter from a method signature.
(Rename Method) – Changes the name of a method while preserving its behavior.
(Rename Class) – Changes the name of a class without altering its structure.
(Rename Package) – Changes the name of a package declaration.
(Rename Field) – Changes the name of a class or instance variable.
(Extract Class) – Moves a group of related fields and methods from an existing class into a newly created class.
(Extract Superclass) – Creates a new superclass to encapsulate shared attributes and behavior from two or more existing classes.
(Inline Method) – Replaces a method call with the method's body.
(Pull Up Method) – Pull a method from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.
(Push Down Method) – Push a method from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Pull Up Field) – Pull a field from a child class (subclass) in the original code to its parent class (superclass) in the transformed version.

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

(Push Down Field) – Push a field from a parent class (superclass) in the original code to one or more of its child classes (subclasses) in the transformed version.
(Inline Class) – Merges a class into another when it is too small or redundant.
(Extract Interface) – Creates a new interface from existing method(s) in a class.
(Move Method) – Relocates a method to a more appropriate class.
(Move Field) – Relocates a field to a more appropriate class.
(Replace Magic Number with Constant) – Replaces a literal number with a named constant.
(Encapsulate Field) – Change a field access from not private to private and create get and set methods.
Instructions:
1. Begin your response with a bullet-point list of the refactoring type(s) applied.
2. Then, briefly justify each identified refactorng with reference to the specific code changes.
3. Only include refactorings from the list above.
4. Be concise but precise in your explanations.

Do not generate explanations unrelated to the given transformation.

- Rename Field

Justification:

- **Rename Field**: The field originally named `title` has been renamed to `title_item`. This change is observed in the `ItemBase` class and the `ItemUpdate` class, both located in `backend/app/models.py`.

Diff

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF

```
--- backend/app/models.py
+++ backend/app/models.py
@@ -67,7 +67,7 @@ class UsersPublic(SQLModel):

 # Shared properties
 class ItemBase(SQLModel):
-    title: str = Field(min_length=1, max_length=255)
+    title_item: str = Field(min_length=1, max_length=255)
     description: str | None = Field(default=None, max_length=255)


@@ -78,7 +78,7 @@ class ItemCreate(ItemBase):

 # Properties to receive on item update
 class ItemUpdate(ItemBase):
```

Explore our developer-friendly HTML to PDF API Printed using PDFCrowd HTML to PDF