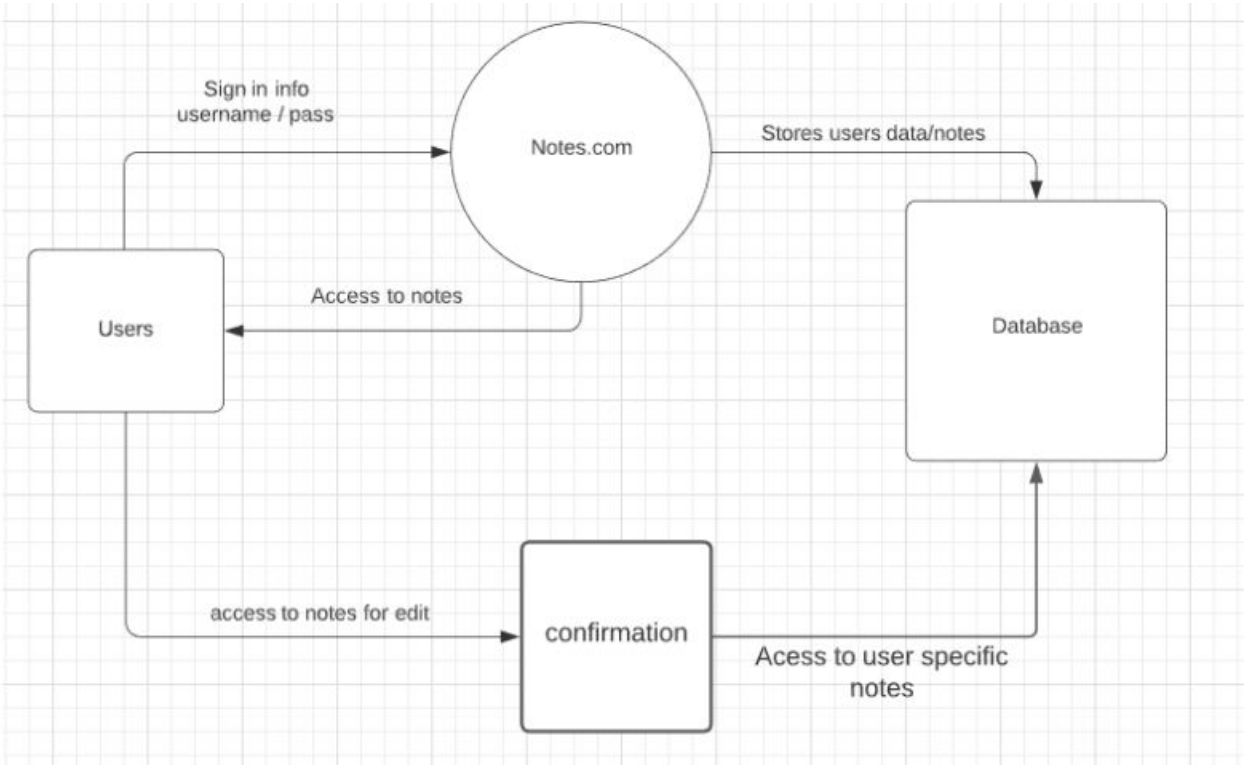
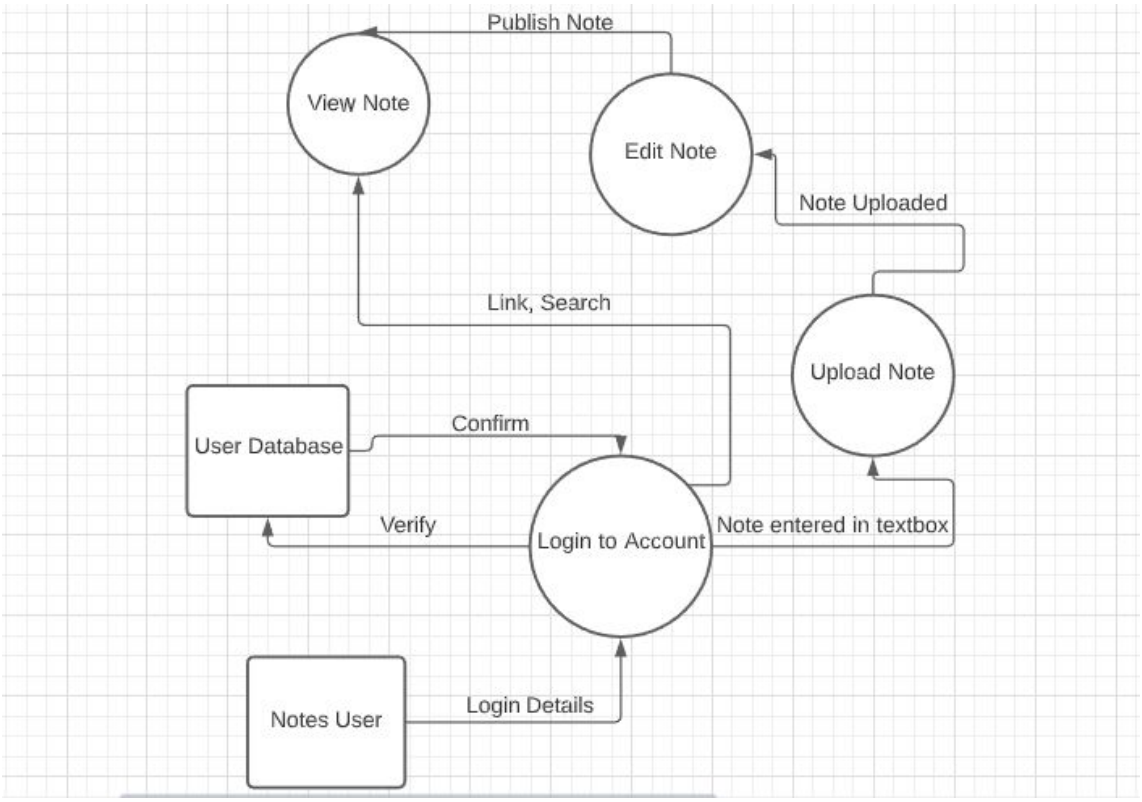
Data Flow Diagram:

Context Diagram:



Level 1 Diagram:



Models and Associations:

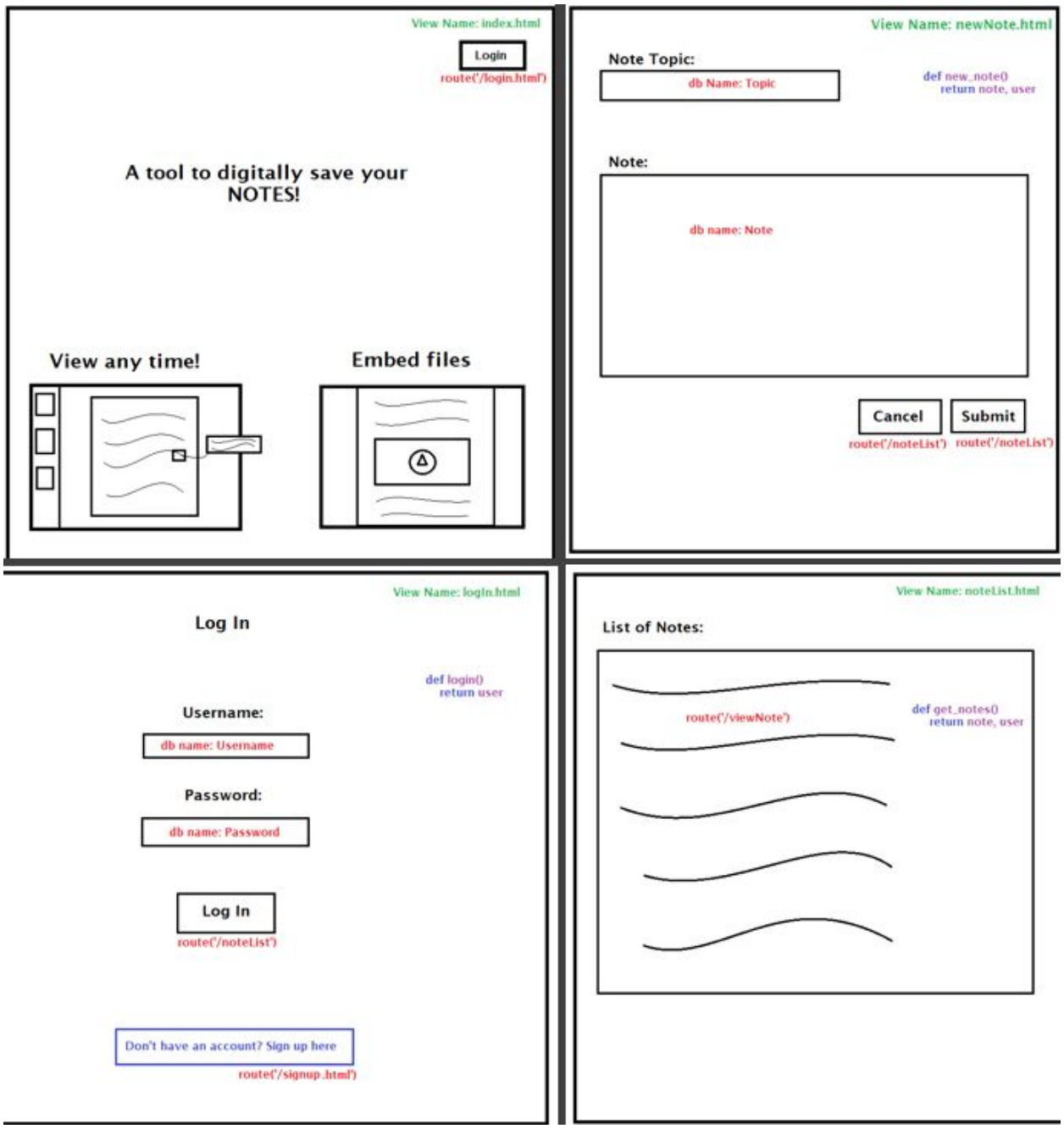
```
class Note(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    title = db.Column("title", db.String(200))
    text = db.Column("text", db.String(100))
    date = db.Column("date", db.String(50))

    def __int__(self, title, text, date):
        self.title = title
        self.text = text
        self.date = date

class editNote(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    title = db.Column("title", db.String(200))
    text = db.Column("text", db.String(100))
    date = db.Column("date", db.String(50))

    def __int__(self, title, text, date):
        self.title = title
        self.text = text
        self.date = date

class deleteNote(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    title = db.Column("title", db.String(200))
    text = db.Column("text", db.String(100))
    date = db.Column("date", db.String(50))

    def __int__(self, title, text, date):
        self.title = title
        self.text = text
        self.date = date

class User(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    name = db.Column("name", db.String(100))
    email = db.Column("email", db.String(100))
    password = db.Column("password", db.String(100))

    def __int__(self, name, email,password):
```

```
        self.name = name
        self.email = email
        self.password = password

class newUser(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    newName = db.Column("name", db.String(100))
    newEmail = db.Column("email", db.String(100))
    newPassword = db.Column("password", db.String(100))
    newPasswordConfirmation= db.Column("Reconfirm password", db.String(100))

    def __int__(self, newName, newEmail, newPassword, newPasswordConfirmation):
        self.newName = newName
        self.newEmail = newEmail
        self.newPassword = newPassword
        self.newPasswordConfirmation = newPasswordConfirmation

class noteComment(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    comment = db.Column("text", db.String(200))

    def __int__(self, comment):
        self.comment = comment

class editComment(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    comment = db.Column("text", db.String(200))

    def __int__(self, comment):
        self.comment = comment

class deleteComment(db.Model):
    id = db.Column("id", db.Integer, primary_key=True)
    comment = db.Column("text", db.String(200))

    def __int__(self, comment):
        self.comment = comment
```

Association:

```
class Note(Base):
        __tablename__ = ‘parent’
        Id = column(integer, primary_key=True)
        children = relationship(“editNote”, “deleteNote”)
class editNote(Base)
        __tablename__ = ‘child’
        id = Column(Integer, primary_key=True)
        note_id = Column(Integer, ForeignKey('note.id'))
class deleteNote(Base)
        __tablename__ = ‘child’
        id = Column(Integer, primary_key=True)
        note_id = Column(Integer, ForeignKey('note.id'))
class user(Base)
        __tablename__ = ‘parent’
        Id = column(integer, primary_key=True)
        children = relationship(“newUser”)
class newUser(Base)
        __tablename__ = ‘child’
        id = Column(Integer, primary_key=True)
        user_id = Column(Integer, ForeignKey('user.id'))
class comment(Base)
        __tablename__ = ‘parent’
        id = Id = column(integer, primary_key=True)
        children = relationship(“editComment”,deleteComment)
class editComment(Base)
        __tablename_ = ‘parent’
        id = column(integer, primary_key=True)
        comment_id = Column(Integer, ForeignKey('comment.id'))
class deleteComment(Base)
        __tablename_ = ‘parent’
        id = column(integer, primary_key=True)
        comment_id = Column(Integer, ForeignKey('comment.id'))
```

Updated Lo-Fi prototypes:




View Name: signup.html
Sign Up
def createUser()
return user
Username
db name: Username
Email
db name: Email
Password
db name: Password
Verify Password
db name: Password
attribute: verifyPass
SUBMIT
route('/noteList')
View Name: comment.html
Add a Comment
def createComment()
return comment, user
Title
db name: Comment, attr: Title
Write your message here...
db name: Comment, attr: Message
Cancel
Submit
route('/viewNote.html')
route('/viewNote.html')
Edit Note
View Name: editNote.html
def editNote()
return user, note
Note Title
db name: UserNote, attr: Title
route('/viewNote')
Save changes
A
H
B
I
U
Embed
Note text
db name: UserNote, attr: Text
View Note
View Name: viewNote.html
def view()
return user, note, comment
Note Title
db name: UserNote, attr: Title
Edit
route('/editNote')
route('/index.html')
Log out
Note text
db name: UserNote, attr: Text
route('/comment')
Add a Comment