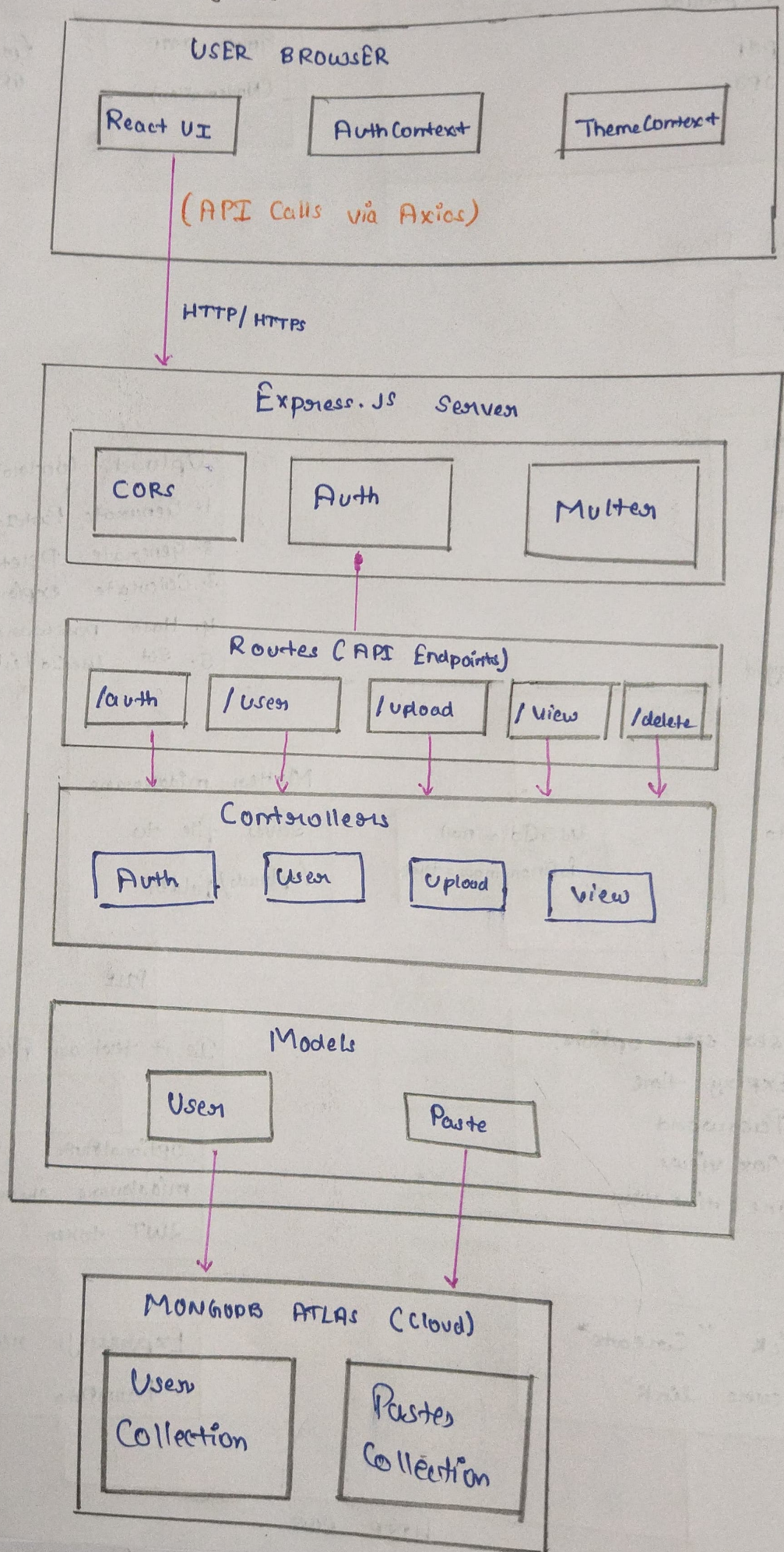


DATA FLOW DIAGRAM

High-level Architecture Diagram



LOCK FILE SYSTEM (server)

/backend/uploads

→ file1.pdf
→ file2.jpg
⋮

External Services

Email SMTP
(Nodemailer)

Google
OAuth 2.0

User Upload Flow

START

User enters
text / selects
file in React
UploadCard

Is user logged
in?

Yes

Add userId to
FormData

userId = null
isAnonymous = true

User sets options!

- Expiry time
- Password
- Max views
- One-time view

Click "Create"
Secure link"

Upload Controller processes:

1. Generate PasteId
2. Generate DeleteToken
3. Calculate expiryTime
4. Hash password (if any)
5. Set UserId / isAnonymous

Multer middleware
saves file to
/uploads/folder

Save text
to
FormData

Is it text or file?

OptionalAuth
middleware checks
JWT token

Express.js receives
FormData

HTTP POST

Create Paste document
in MongoDB

```
{  
  pasteId,  
  userId,  
  isAnonymous,  
  type,  
  content/fileUrl,  
  expiresAt,  
  passwordHash,  
  maxViews,  
  deleteToken  
}
```

MongoDB Insert

MongoDB Atlas saves
paste Returns _id

Controller returns response:

```
{  
  id: pasteId,  
  url: /view/pasteId,  
  deleteToken,  
  expiresAt  
}
```

HTTP response.

React receives response and
displays success

Is user authentication?

NO
(ANONYMOUS)

Save to
browser's local storage!

```
{  
  pasteId,  
  deleteToken,  
  expiresAt  
}
```

Paste already
linked to
userId in DB
No need for
local storage.

Logged-In

END

Display link to user
with copy button