Cursor Prompt: Student Assignment Submission Portal (MVP)

Context

We want to allow students to submit their assignments directly from the assignment splash page within a course. After submitting, the button changes to “Resubmit.” Each submission (PDF, etc.) should be stored and tracked, but students only see/download their latest submission. Teachers should later be able to see the full submission history.

Step 1: Schema/Prisma Updates

- **Create/extend a `Submission` model in Prisma:**
    - Each submission should be linked to:
        - The assignment (`assignmentId`)
        - The student/user (`studentId`)
        - The uploaded file (PDF, etc.)
        - A timestamp (createdAt)
        - An auto-incrementing order (to track submission # for a given student & assignment)
    - Optionally: Add any “status”/metadata you want (e.g., late, on time)

Schema Example:

prisma
CopyEdit

```
model Submission {

  id           String   @id @default(uuid())

  assignment   Assignment @relation(fields: [assignmentId],
references: [id])
```

```
  assignmentId String

  student      User      @relation(fields: [studentId], references:
[id])

  studentId    String

  fileUrl      String

  createdAt    DateTime @default(now())

  order        Int

}
```

-
    - *Note*: Adjust naming/types for your schema

Step 2: API Endpoints

- **POST /assignments/{assignmentId}/submissions**
    - Auth: Only accessible by logged-in students in the course
    - Accepts file upload (PDF)
    - Stores the submission, increments the order for that student/assignment combo
- **GET /assignments/{assignmentId}/submissions/self**
    - Returns latest submission for the logged-in student (for “View Submission” button/iframe)
- **GET /assignments/{assignmentId}/submissions/history/self**
    - (Optional for now) Returns all submissions by student for that assignment
- **GET /assignments/{assignmentId}/submissions** *(for teacher dashboard later)*
    - Returns all submissions for that assignment (grouped by student)

Step 3: Frontend UI Logic

- On the assignment splash page (student view):
    - **If no submission exists:**
        - Show “Submit Assignment” button
    - **If a submission exists:**
        - Show “Resubmit Assignment” button (replace file on upload)
        - Show “View Submission” button to open latest PDF in the existing iframe viewer
    - Only the *latest* submission is visible to the student, but all are stored in the backend

---

Step 4: Submission Handling Logic

- When a student submits (or resubmits):
    - Upload the new file and create a new Submission record (don’t overwrite old ones)
    - Increment the submission order (can also be based on `createdAt`)
    - The old submissions for this student/assignment remain in the database but are not visible to the student
    - In the future, teachers will be able to view/download all submissions for audit/plagiarism/etc.

Step 5: Future-proofing

- Design the endpoints and schema so that adding feedback/grading/comments per submission later will be easy (e.g., link submission to feedback/grade records in future releases)

- Leave hooks in UI for showing “Submission History” to students later if you ever want

---

What to Tell Cursor Specifically

- Refactor/add a Submission model as above

- Implement API endpoints for submit/resubmit and getting the latest submission

- Add submission upload button and “View Submission” (iframe) button to the student assignment splash page UI

- Ensure permissions: Only students in the course can submit; only their latest submission is shown to them

- When resubmitting, do not delete old submissions—just create a new one and show the latest

- Keep code modular so teacher features (submission history, grading) are easy to add