WORK UPDATE

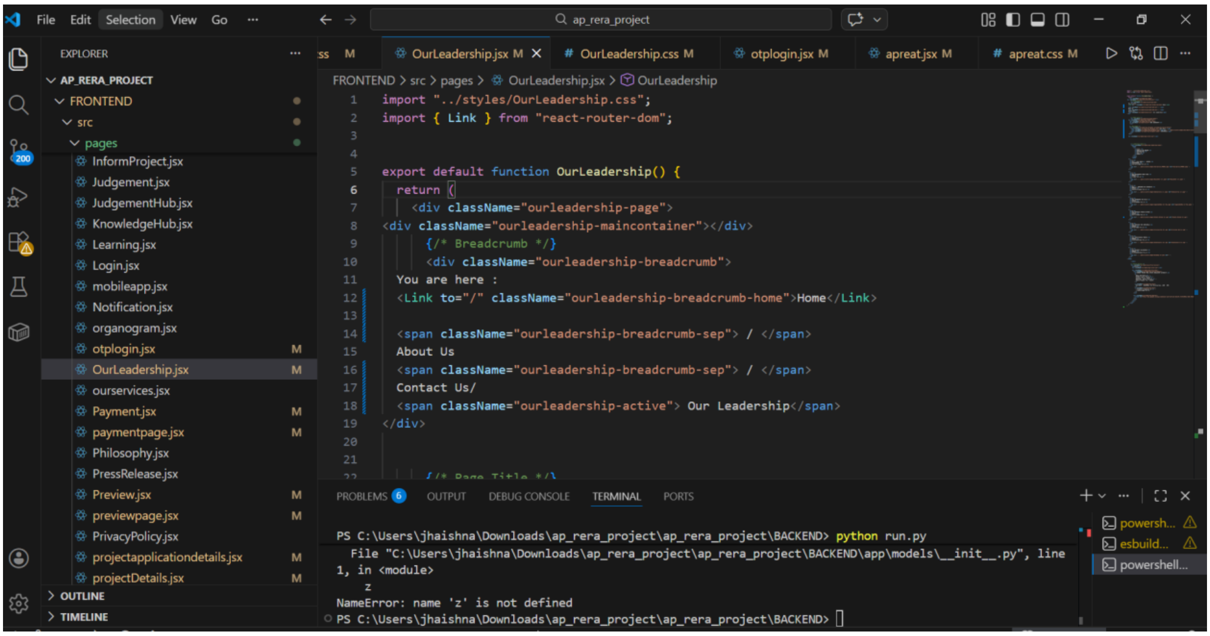
NAME:D Sri Ramya

Date:9-2-2026

1. Overview of Work Performed

During this development phase, I worked extensively on the Extension Application module to ensure smooth user flow, proper frontend–backend integration, and compliance with functional and validation requirements. The primary focus areas included implementing the extension process UI, handling supporting document uploads, validating user inputs, fixing issues reported during testing, and ensuring data consistency across screens and APIs.

The work was carried out with close attention to functional requirements, test cases, and feedback received from the testing team. All changes were tested thoroughly to avoid regressions and to ensure system stability.



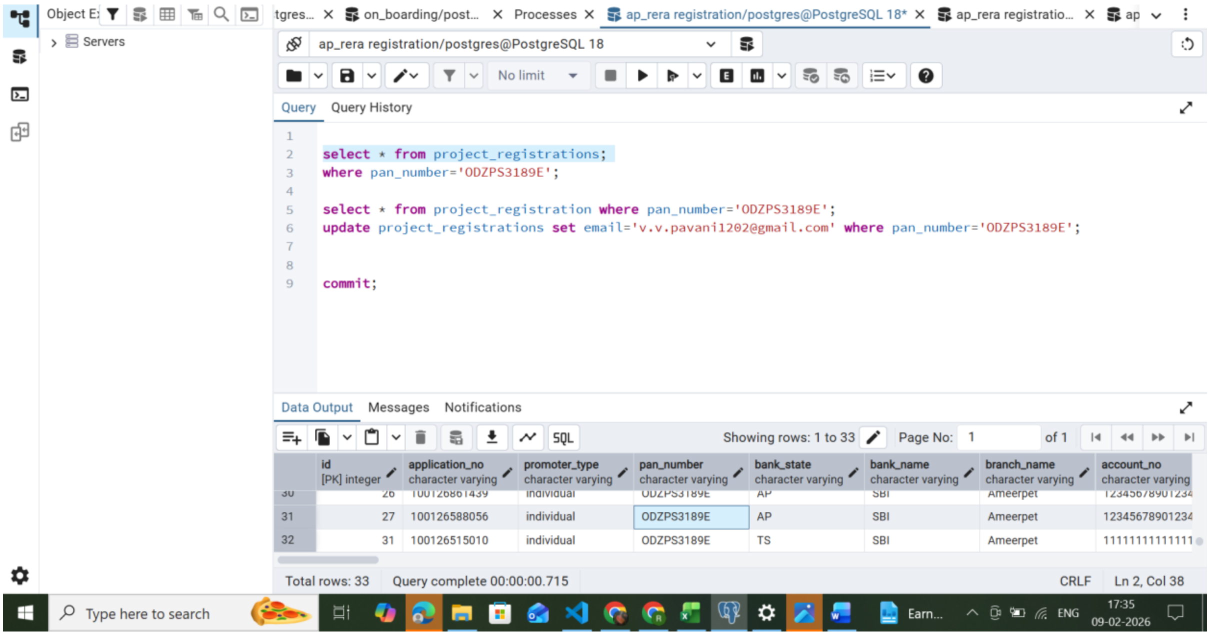
```
import "../styles/OurLeadership.css";
import { Link } from "react-router-dom";


export default function OurLeadership() {
  return (
    <div className="ourleadership-page">
<div className="ourleadership-maincontainer"></div>
      {/* Breadcrumb */}
      <div className="ourleadership-breadcrumb">
  You are here :
  <Link to="/" className="ourleadership-breadcrumb-home">Home</Link>

  <span className="ourleadership-breadcrumb-sep"> / </span>
  About Us
  <span className="ourleadership-breadcrumb-sep"> / </span>
  Contact Us/
  <span className="ourleadership-active"> Our Leadership</span>
</div>



      {/* Page Title */}
```

```
PS C:\Users\jhaishna\Downloads\ap_rera_project\ap_rera_project\BACKEND> python run.py
  File "C:\Users\jhaishna\Downloads\ap_rera_project\ap_rera_project\BACKEND\app\models\__init__.py", line
1, in <module>
    z
NameError: name 'z' is not defined
PS C:\Users\jhaishna\Downloads\ap_rera_project\ap_rera_project\BACKEND> 
```

2. Extension Application Form Development

I implemented the Extension Application form end-to-end, ensuring that project data is correctly carried forward from the previous screen using routing state. The application number, project name, project ID, and existing validity dates are auto-populated and displayed as non-editable fields to avoid user manipulation.

Additionally, new validity dates are captured using proper date input fields with mandatory validation. These inputs strictly follow the requirements specified in the extension process documentation. The UI layout was aligned with existing design standards to maintain consistency across the application.



```
select * from project_registrations;
where pan_number='ODZPS3189E';

select * from project_registration where pan_number='ODZPS3189E';
update project_registrations set email='v.v.pavani1202@gmail.com' where pan_number='ODZPS3189E';


commit;
```

Showing rows: 1 to 33 Page No: 1 of 1

	id [PK] integer	application_no character varying	promoter_type character varying	pan_number character varying	bank_state character varying	bank_name character varying	branch_name character varying	account_no character varying
30	26	100126861439	individual	ODZPS3189E	AP	SBI	Ameerpet	12345678901234
31	27	100126588056	individual	ODZPS3189E	AP	SBI	Ameerpet	12345678901234
32	31	100126515010	individual	ODZPS3189E	TS	SBI	Ameerpet	11111111111111

Total rows: 33 Query complete 00:00:00.715 CRLF Ln 2, Col 38

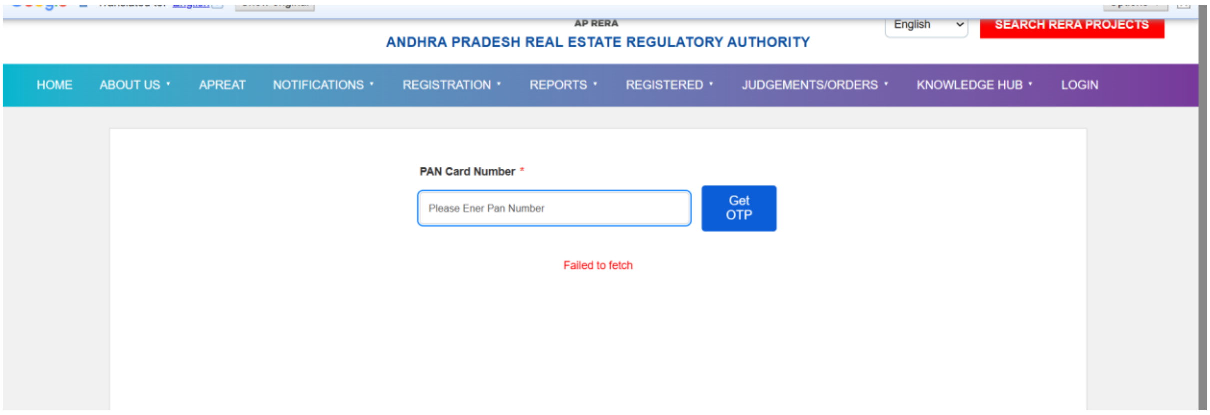
---

3. Supporting Documents Upload Implementation

A major part of the work involved implementing the Supporting Documents section. The following documents were handled as mandatory uploads:

1. Representation Letter explaining the reason for delay
2. Form B with revised completion dates
3. Consent letter from the allottees
4. Form E for renewal
5. Change Request in Form P4
6. Extension proceeding granted by the local authority

Each document field was mapped correctly to the backend API keys to ensure seamless data submission.



PAN Card Number *

Please Ener Pan Number

Failed to fetch

---

4. File Upload Validation Enhancements

Based on functional requirements and test cases, I implemented strict file upload validations. Each upload field now allows **only one file per field**, preventing users from selecting or uploading multiple files in the same input.

In addition:

- Only PDF files are allowed.
- If a user attempts to upload a non-PDF file, an error message is displayed.
- If multiple files are selected (via drag and drop or other means), the system blocks the action and shows a clear validation message.
- File selection is reset automatically on validation failure to avoid incorrect submission.

These validations ensure compliance with test case requirements and prevent incorrect data from being sent to the backend.

---

5. Frontend and Backend Integration

I ensured that the frontend request payload matches the backend API expectations exactly. All form fields and file uploads are appended correctly using FormData. Mandatory fields are validated before submission, and appropriate alerts are shown if any required input or document is missing.

After preparing the payload, the data is submitted to the backend extension application API. Successful responses trigger user navigation to the payment page, while failures are handled gracefully with proper error messaging.

---

6. Bug Fixes and Testing Support

I addressed multiple issues reported during the testing phase. One significant issue involved missing project records due to an INNER JOIN in the backend query, which was filtering out valid projects. After analysis, the query was updated to use a LEFT JOIN, ensuring all relevant project records linked to the PAN are retrieved correctly.

I also resolved frontend–backend integration issues caused by missing mandatory fields in API requests. After aligning the request payload with backend validation rules, the issues were resolved successfully.

---

7. Testing and Verification

All implemented changes were tested thoroughly, including:

- Single file upload validation per field
- PDF-only file enforcement
- Mandatory field validation
- Successful and failure API response handling
- Navigation flow after successful submission

Testing was performed to ensure that all functional and negative scenarios behave as expected.